AN IMPROVED ALGORITHM FOR SEQUENTIAL INFORMATION-GATHERING DECISIONS IN DESIGN UNDER UNCERTAINTY

A Thesis

by

TYLER RAYMOND HALBERT

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Richard Malak
Co-Chair of Committee,	Darren Hartl
Committee Member,	Douglas Allaire
Head of Department,	Andreas A. Polycarpou

August 2015

Major Subject: Mechanical Engineering

ABSTRACT

In engineering decision making, particularly in design, engineers must make decisions under varying levels of uncertainty. While not always the case, oftentimes one of the options available to an engineer is the ability to gather information that will reduce the uncertainty. With the reduced uncertainty, the engineer then returns to the same decision with more information. This sequential information-gathering decision problem is difficult to analyze and solve because the engineer must predict the value of gathering information in order to determine if the value outweighs the cost of the resources expended to gather the information. In practice, heuristics, intuition, and deadlines are often used to decide whether or not to gather information. A more complete and formal approach for quantifying the value of gathering information would benefit engineers in design decision making.

Recent work proposed that a Partially Observable Markov Decision Process (POMDP) is an appropriate formalism for modeling sequential information-gathering decisions. A POMDP appears capable of capturing the salient features of such decisions. However, existing POMDP solution algorithms scale poorly with problem size. This thesis introduces an improved algorithm for solving POMDPs that takes advantage of certain characteristics inherent to information-gathering decision problems. The new algorithm is orders of magnitude faster and also is capable of handling specific problem parameters that existing methods cannot. The improvement is shown with a detailed case study, where the case study also performs a comparison of using the POMDP formalism

for solving information-gathering decision problems to widely known approximate methods, such as Expected Value of Information methods. The study demonstrates that the use of the POMDP formalism, along with the improved algorithm, provides a valuable method for solving certain information-gathering decision problems.

ACKNOWLEDGEMENTS

I would like to thank my committee co-chairs, Dr. Richard Malak and Dr. Darren Hartl, and my committee member, Dr. Douglas Allaire, for their guidance and support throughout the course of this research.

Thanks also go to my friends and colleagues and the department faculty and staff for making my time at Texas A&M University a great experience. I also want to extend my gratitude to the National Science Foundation, which provided funding for my research and education.

Finally, thank you to my mother and father for their encouragement, and most importantly to God for giving me the ability to pursue such a difficult course of study.

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

LIST OF TABLES

Page

1. INTRODUCTION

1.1 Uncertainty in Engineering Design

In engineering design, engineers often are required to make difficult decisions under varying levels of uncertainty. This uncertainty is due to the fact that engineers do not possess perfect knowledge of the world, and as a result, their design. There will always be some level of uncertainty about how a design will perform compared to predictions, no matter how confident and knowledgeable the engineer may be. Uncertainty is generally classified into two categories: epistemic and aleatoric [1, 2]. Aleatoric uncertainty is uncertainty due to natural random behavior in a physical system. Epistemic uncertainty on the other hand is uncertainty that is a result of lack of knowledge or information. Aleatoric uncertainty, in the general sense, is considered to be irreducible, while epistemic uncertainty can be reduced through increased information. For the purposes of this thesis, the use of the word "uncertainty" will refer to epistemic uncertainty.

Examples of uncertainty in engineering design decisions include the capabilities of a particular technology, the interaction between components in a system, and the performance of a component or design in a specific environment. Even after the design portion of a project is completed, imperfections and tolerances in the manufacturing or building process may have an effect on performance, thus contributing to the uncertainty. At a higher level of an engineering project, there may be uncertainties in the

design costs, schedule and budget. All of these types of uncertainty affect the decision maker (DM) to some extent in engineering design, whether the effect is apparent or not.

The study of design under uncertainty is well-developed and has generated many different methods for managing uncertainty. These methods take a wide range of approaches to incorporating uncertainty into engineering design. Table 1 presents a brief summary of the uncertainty management methods to be discussed in this section.

Table 1. Summary of uncertainty management methods.

Method	Uncertainty Management Approach
Risk Management	Minimizes likelihood of failure of design
Reliability-based Design Optimization (RBDO)	Places probabilistic constrains on failure modes of design
Robust Design	Seeks a design insensitive to the uncertainty
Real Options	Builds in options to design that can account for uncertain outcomes after realization
Flexible Design	Seeks a design that is flexible and can adapt to uncertain outcomes
Set-based Design	Designs multiple concepts in parallel and delays selection of final design to reduce uncertainty
Decision-based Design	Maximizes expected utility of design over uncertain outcomes

One category of uncertainty management methods, known as risk management, manages uncertainty by guiding the DM to choose a design that minimizes the risk in the final design, where risk is the likelihood of occurrence of a negative effect on the project objective(s) [3]. This is a very common approach where its premise is to place the focus

on guarding against failure in the design process [4-6]. A common and basic way of achieving this is through the use of a safety factor, which leads to designs that essentially are over-designed in order to avoid failure and reduce risk.

A method similar to risk management is reliability-based design optimization (RBDO). In RBDO, the DM seeks the optimal design characterized by a low probability of failure and minimum cost [7, 8]. This optimization is generally achieved through minimizing cost subject to probabilistic constraints on critical failure modes. In this sense, it is similar to risk management because both methods seek to minimize the probability of failure. Another related method of uncertainty management is robust design. The fundamental principle of robust design is to create a design that is insensitive to the associated uncertainty [9]. By doing so, the uncertainty in the performance of the final design is significantly reduced. A widely used robust design method is Taguchi's robust design, or Taguchi methods, which is a set of statistical techniques that apply to the entire process of developing and manufacturing a robust design [10, 11].

The field of real options manages uncertainty by building in options to the design that account for different uncertain outcomes [12]. For example, a project manager may be uncertain in the total capacity required for a parking garage. An example of a real option is designing the garage such that additional floors can be added at a later date. By doing so, he is designing the garage such that it can adapt to the uncertain outcome of required parking capacity. In this way, portions of a design, or even an entire design itself, may go unused. Real options provide the design the ability to expand or adapt to

uncertain outcomes. Because of this, it is considered a method of flexibility in design, which seeks to create designs that can adapt to uncertain outcomes [13, 14]. Because of this objective, flexibility in design can be considered similar to robust design, because a design that is flexible to different uncertain outcomes is essentially a method of creating a design that is insensitive to the uncertainty [14].

In contrast to these methods, set-based design takes a different approach to managing uncertainty [2, 15]. Using set-based design, a DM groups design alternatives into sets which are defined by a general concept. The focus is placed on eliminating inferior concepts, as opposed to selecting the best concept, carrying as many viable concepts as possible through the different stages of the design process. This allows the DM to delay choosing a final design and develop multiple alternative design concepts in parallel until the concepts have been more fully developed and the uncertainty in the outcomes of the different concepts is reduced. Set-based design has been implemented in industry by Toyota and is considered a significant factor in their success as a leader in product quality [16].

Decision analysis takes a normative approach to making decisions under uncertainty, in which the DM chooses the decision alternative that has the highest expected utility [17, 18]. The DM is uncertain in the specific outcome that will result from choosing each of the available decision alternatives. For every decision alternative, each possible outcome has a defined utility (i.e., the utility that this outcome brings to the DM) where the uncertainty is defined by the probability that each outcome occurs. The expected utility of each decision alternative is then calculated by the summation of

all the possible outcomes weighted by each outcome's probability of occurrence. This allows the DM to rank decision alternatives by expected utility and choose the alternative that has the highest expected utility.

Using decision analysis as its foundation, decision-based design provides a framework for engineers to make design decisions with the objective of maximizing the expected utility of the design [19-22]. In decision-based design, the DM assigns a utility to each design alternative, generally as a function of (or at least dependent on) the attributes of the design. In the presence of uncertainty, a utility is defined for each uncertain outcome that results from choosing a design. This allows engineers to make decisions that focus on the attributes of a design only insofar as they affect the design's utility.

All of these methods provide different approaches to managing uncertainty in design; however, in general, most are implemented with the uncertainty considered to be constant for a given problem. This is not necessarily a theoretical limitation for all methods; however, most implementations do not directly treat the uncertainty as a variable that can be altered by the DM. In other words, they consider only the design artifacts as alternatives for the DM to choose from and do not directly present an alternative for modifying the uncertainties, such as by gathering relevant information about the uncertain parameter(s) of interest. For example, an engineer may conduct a mechanical test on a structural component that is relevant to the design decision and then return to the original decision with better information. A reduction in uncertainty allows the DM to choose an alternative with greater confidence that their decision will achieve

the specific desired outcome. This reduction in uncertainty does not come without a cost from gathering the needed information. Costs can include both the cost of physically performing the tasks required to gather information as well as the delay in schedule caused by postponing the final decision. As such, engineers must balance the cost of the information-gathering action with the benefit of increased informatione about the problem and a likely reduction in uncertainty.

In fact, most implementations ignore many of the decisions in the design process, including gathering information. Instead, they place the emphasis on choosing the best design from a set of alternatives. Recognizing that formulating design problems in this way is too narrow, Thompson and Paredis established a decision-based design perspective that analyzes the design process in an enterprise context [23]. This perspective allows the DM to consider the resources expended in the design process in addition to the design artifacts alone. Particularly, the authors demonstrate the importance of considering the resources expended to gather information relevant to design decisions. This is relevant in nearly all engineering design fields, where gathering information is an important part of the design process.

It is often difficult to determine if an information-gathering action is valuable, particularly because the DM usually does not know exactly how valuable the information will be before it is obtained. Some of the uncertainty management methods previously presented indirectly incorporate gathering information. For example, in set-based design, by delaying the selection of the final design until later stages in the design process, the DM can incorporate gathered information as the design project progresses.

Additionally, with real options, the DM can build in an option to the design that can be exercised depending on information gathered after the design has been finished. In the case of the parking garage example considered earlier, the DM may learn after building the garage that the required capacity is higher than anticipated. By building in the option to add extra floors, the DM can now act based on the new information about the garage capacity. However, none of these approaches directly include options for the DM to gather information about any uncertain parameters or consider the value that could be added to the project by gathering information.

As a formal quantification of the value of gathering information using decision analysis as its foundation, value of information (VoI) approaches compare the value of the decision with and without additional information [18, 24, 25]. There are several examples within the design community of applying VoI [26-30]; however, these applications all suffer from one of the following drawbacks: they lean on the principles of the theory for direction but do not apply the theory rigorously to calculate the value of information, or they do not directly consider the cost of gathering the information. Both the costs and value of gathering information must be considered in order to make a decision that maximizes expected value.

As a means of achieving this, Expected Value of Information (EVI) approaches use VoI theory to explicitly calculate the expected value of gathering information [18, 24]. This is done by taking the difference between the expected value of choosing a design artifact with and without additional information. There are two main approaches for doing this: Expected Value of Perfect Information (EVPI) and Expected Value of

Sample Information (EVSI) [17]. EVPI assumes that the DM can gather perfect information about the decision problem (i.e., gathering information eliminates the uncertainty). This is usually a poor assumption because engineers rarely have access to perfect information. As such, EVPI provides an upper bound on the expected value of gathering information by providing the best case scenario of receiving perfect information. Because of this, EVPI is often used in engineering design decisions informally, heuristically, or only as a preliminary calculation (see [26, 31, 32] for specific applications).

By comparison, EVSI considers imperfect sources of information, sources from which the gathered information may be inaccurate. By incorporating an imperfect source of information, EVSI allows the information-gathering action to be chosen more than once. For example, in the design scenario for the case study, the DM may pursue mechanical testing of the material multiple times before the uncertainty is sufficiently reduced. EVSI directly accounts for this and calculates the expected value of gathering information as well as the number of "samples" that the DM should gather (the number of times they should pursue the information-gathering action). As such, EVSI is a common method used to determine the optimal sample size for a study [33]; it is particularly popular in the medical field where it is used to determine the number of clinical trials for new medical treatments [34-37].

Although EVSI accounts for imperfect sources of information, it does not fully represent information-gathering decisions because it does not capture the sequential aspect of information-gathering decisions. This is because the EVSI approach (and EVPI

as well) essentially treats the information-gathering decisions as a separate sub-decision [23]. The DM must first decide whether or not to gather information (and how much to gather) and then proceeds to the design artifact decision. This process is illustrated in Figure 1.

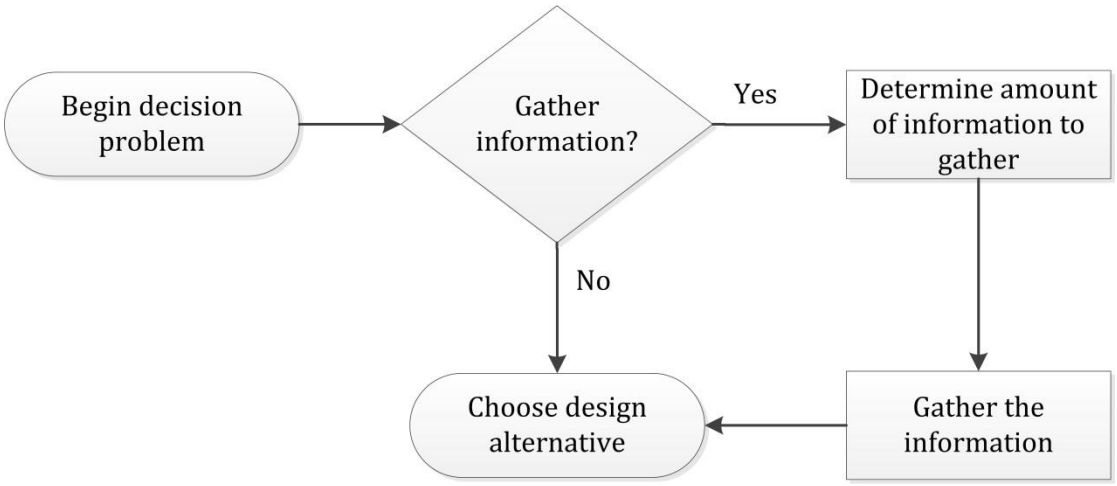

Figure 1. Description of EVSI decision process.

This is not an accurate representation of reality, where after gathering imperfect information, the DM can often choose to gather information again. This can be repeated many times until the DM's uncertainty in the outcomes of the design artifact decision is sufficiently reduced. In the EVSI approach, however, the DM must choose how much information to gather at the onset of the problem. Thus, in the calculation of the expected value of gathering information, it does not explicitly account for the possibility that, after gathering information, the DM may revisit the decision and choose whether not to gather more information based on the information they have already received. In reality, the

DM would choose to gather information again if they expected to gain additional value, but this possibility is not considered in EVSI. Thus, the EVSI calculation provides a lower bound on the value of gathering information because it neglects the option for the DM to gather information sequentially, which can only increase the expected value.

In addition to EVI approaches, there are several fields of research which deal with optimum data collection. One such field is maximum entropy sampling [38]. As the name suggests, this method makes judgments based on information entropy [39]. As a result, this makes explicitly assessing the tradeoff between the value of information and the associated costs difficult, because a relation must be made between information entropy and value. Another method with a similar limitation is the use of a knowledge-gradient policy. This can be applied to sequential information collection in the presence of multiple sources of uncertainty, where a knowledge-gradient policy maximizes the increase in expected value of information of each sequential information gathering action [40]. However, this method generally considers problems where the number of times the DM may gather information is fixed and the emphasis is placed only on choosing which sources of uncertainty to gather information about.

Another method of optimum data collection is optimal Bayesian experimental design, where the DM seeks to find a set of experiments, or experiment parameters, that provide the most information about targeted uncertain parameters [41]. For example, simulation-based models can be used to generate this set of experimental parameters using an expected utility framework [42]. The set of experimental parameters can also include the number of times the experiment is conducted. As with the other optimum

data collection methods presented, the costs of executing the experiment are generally not directly considered in optimal experimental design.

Other optimum data collection methods include optimal sample size and Bayesian optimal stopping, which provide approaches for determining when to stop gathering information [33]. All of these approaches only partially (if at all) model the sequential nature of information-gathering problems in combination with a direct incorporation of the associated costs, which is summarized in Table 2.

Table 2. Summary of information-gathering analysis methods.

Method	Sequential Nature?	Cost Consideration?
Expected Value of Information (EVI)	No	Yes
Maximum Entropy Sampling	Partial	No
Knowledge-Gradient	Partial	No
Bayesian Experimental Design	Partial	No
Optimal Sample Size	No	Yes
Bayesian Optimal Stopping	Partial	Yes

This is largely due to the fact that several of these approaches are directed toward information-gathering problems where the DM has already decided to gather information or is likely to gather information a large number of times. Because of this, the simplified problem representation is generally sufficient, because the DM is not trying to decide whether or not to gather information, only how much to gather. Furthermore, due to the computational complexity required of such methods, most

applications apply to only a subset of information-gathering problems, such as problems with only one information-gathering action or where a normal distribution can be assumed [43, 44].

1.2 Design under Uncertainty Example with Information-Gathering Actions

From a design project perspective, a DM often may not know whether or not to gather information about any relevant uncertain parameters, or even which specific parameter(s) to gather information about. As a concrete example of such a design project that considers information-gathering and its associated resource expenditures, the design project scenario used for the case study in this thesis is presented here. While the purpose and results of the case study are presented in later sections, the design scenario is described here because it provides a concrete example of the relevance of information-gathering in engineering design.

The context of the case study is the design of a deployment mechanism for an origami-inspired solar array to be used on a space mission by NASA. This solar array, proposed in a joint effort between Brigham Young University and NASA's Jet Propulsion Laboratory [45] and shown in Figure 2 (Copyright 2013 by Daily Herald), provides a large surface area for harvesting solar energy but folds into a small footprint for launch into space.

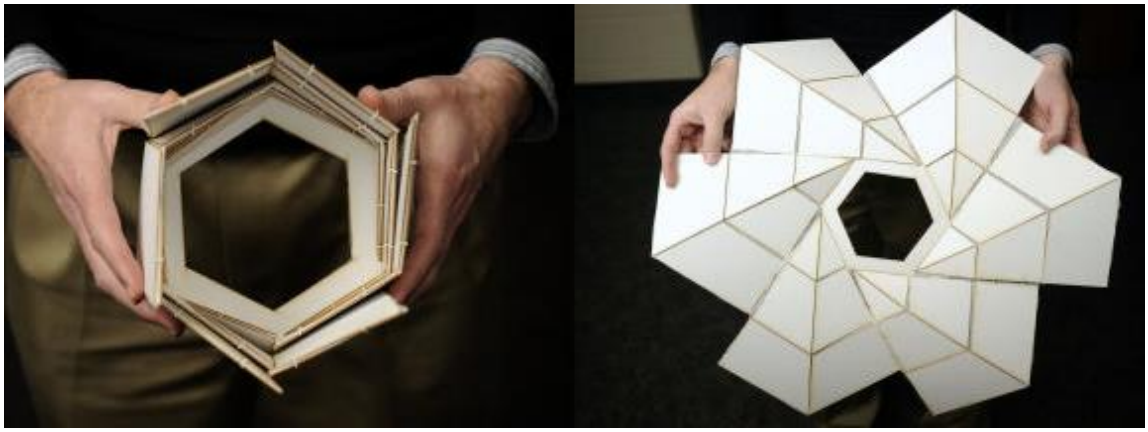

Figure 2. Origami-inspired solar array in both the stowed (left) and open (right) configurations [46].

Considering Figure 2, the solar array must be deployed in such a way that it unfolds from the stowed configuration (left) to the open and flat configuration (right). This case study considers the scenario where the design team has narrowed the choice of concepts to two different deployment mechanism concepts (denote them A and B for now), where the design problem is to select one of the two mechanisms and optimize the sizing of the corresponding components appropriately. For this case study example, the sizing parameters considered are reduced to just one parameter for each design, which is the length of each concept (this will be explained in more detail in Section 4). Thus, the DM must choose one of the two available design concepts for the actuation mechanism as well as specify the length of the chosen mechanism.

However, for each concept, the DM is uncertain in a specific material property that directly affects the ability of the concept to achieve the desired unfolding. For each uncertain material property, there is an associated mechanical testing option which can

be executed in order to gather information about and modify the uncertainty. The mechanical testing options can each be executed as many times as the DM deems necessary, but a specific cost is incurred with each execution. A rational DM only desires to gather information if they think that the information provided will be worth the cost of acquiring it. Thus, the DM is presented with a decision problem in which they must choose whether or not to gather information (and which uncertain parameter to gather the information about) before choosing and optimizing the best available design concept.

Depending on the level of uncertainty, it is feasible that the DM chooses not to gather any information and design the solar array actuation mechanism at the current level of uncertainty. This may occur when the uncertainty in the material properties is very low, for example when the DM has significant experience working with the specific material. However, when this is not the case, the DM must decide whether gathering information about either of the uncertain material properties is worth the cost. If the DM chooses to gather information, they must again choose what to do once the information has been gathered. It is reasonable that, depending on the accuracy of the information received, that more information should be gathered before choosing the design concept. For example, if the DM chooses to gather information about only one of the two material properties, it is possible that based on the resulting information, the DM now wishes to gather information about the other material property.

Overall, this design scenario demonstrates that there may be multiple sources of information in a design project, and that a DM must be able to decide whether or not to

pursue these options, or simply choose the best available design at the current level of uncertainty.

1.3 A New Approach to Modeling Information-Gathering Decisions

In order to most accurately model an information-gathering decision problem, such as the design scenario presented in the previous section, the sequential nature should be fully captured. This is particularly important because the DM does not know exactly what information they will receive before gathering it. If a DM chooses to gather information, the subsequent decision that they will make, after receiving the information, will be dependent upon the information received. For example, consider the case study defined previously where the DM is very uncertain in both material properties: if the DM chooses to gather information about the material property for design concept A, this information will likely affect whether or not they choose to also gather information about the uncertain property of design concept B. If the information they receive reveals that the material property for A almost certainly makes this concept superior to B no matter what the uncertain material property for B truly is, they will choose not to gather information about the material property for B because concept A is superior. However, if the information reveals that the material property for A is undesirable, the DM would then be more inclined to gather information about the material property for B because they would like to know if concept B is better than A. This demonstrates that the sequential nature of information-gathering decisions is important, because future

decisions (in the case of gathering information) can be dependent upon the specific information that is received.

The only way to fully capture all aspects of the information-gathering decision problem is to explicitly model the sequential nature of the problem. As noted by Thompson and Paredis [23], the most basic method of achieving this is to represent the problem using the decision tree structure, where every possible sequence of decisions is explicitly considered by using decision analysis in a sequential context. However, by accounting for all possible sequence combinations, the size of the decision tree quickly becomes very large for anything but small, compact problems. As such, computing the solution to the decision tree using backward induction is quite often computationally prohibitive, particularly when the uncertain parameter of interest is a continuous parameter. This is demonstrated in [23] with a pressure vessel design example, where the assumption of a normally distributed uncertain parameter is necessary to simplify the computation of the solution.

In general, all of the aforementioned methods for approaching information-gathering decisions (see Section 1.1), with the exception of the decision tree structure, make simplifying assumptions in order to avoid the computational complexity associated with solving an explicit representation of the problem, such as through a decision tree. These methods, while not explicit (i.e., they do not rigorously incorporate all aspects of the information-gathering problem in a mathematical framework), do hold value for engineers as they help guide information-gathering decision-making, albeit heuristically.

However, these methods lack the ability to consistently and accurately determine the best course of action for design engineers.

Furthermore, solving the explicit representation in decision tree form is often computationally intractable. As such, a more efficient method of explicitly modeling this problem would provide a much more valuable and complete way of making information-gathering decisions such that information is only gathered when it is expected to increase the value of the design project.

As a proposed method of explicitly modeling the sequential nature of information-gathering decisions, Hsiao and Malak introduced the use of Partially Observable Markov Decision Processes (POMDPs) [47]. They present and qualitatively discuss the benefits of the POMDP formalism over other methods. Specifically, posing an information-gathering decision problem as a POMDP explicitly considers the sequential aspect of information-gathering decisions as well as the costs and benefits. This representation can fully capture all aspects of the decision tree structure, but it does so using a compact format that avoids the combinatorial growth of the decision tree representation. In addition to the compact problem representation, the solution to a POMDP provides not only the current best action for the DM to take, but also provides the subsequent best action to take at future decisions (in the case that an information-gathering action was chosen). It is worth noting that it has yet to be proven that every decision tree can be represented using a POMDP. However, the opposite has been proven true, in that every POMDP can be represented by a decision tree [48]. This thesis

considers specific information-gathering decision problems that can be represented as POMDPs, which are described in detail in Section 2.

The use of POMDPs relies on the repeating structure of general information-gathering problems to create the compact representation: after gathering information, the DM returns back to a structurally similar decision but with updated uncertainty (this is described in detail in Section 2). This repeating structure dramatically simplifies the problem representation, which allows the POMDP representation to explicitly solve more complicated information-gathering problems than the decision tree structure by using existing POMDP solution methods. Several POMDP solution techniques exist; however, information-gathering problems represented as POMDPs can still become computationally expensive to solve as the complexity of the problem grows. This phenomenon is not unique to information-gathering problems, as POMDPs have been used to model decisions in other fields such as robot navigation [49, 50] and medical diagnosis [51].

While POMDP solution methods are improving, current methods are not well-suited to the potential large size of information-gathering problems [47, 52-54]. However, there are specific features inherent to information-gathering decision problems that can be leveraged. In particular, information-gathering decision problems form a small subset of sequential decision problems that can be represented using the POMDP formalism. This is illustrated in Figure 3 using a Venn diagram.

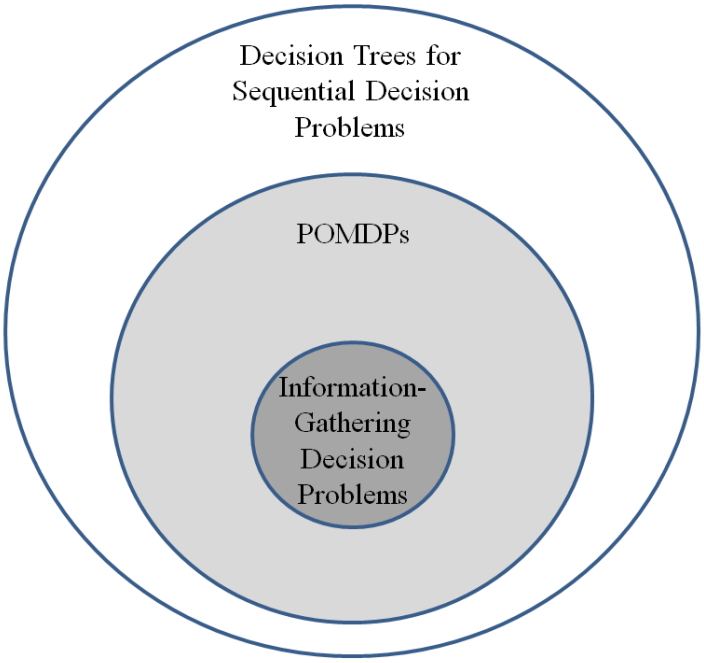

Figure 3. Venn diagram of information-gathering decision problems within decision trees and POMDPs.

This subset has certain characteristics that reduce the complexity of representing and solving this particular problem as a POMDP. By taking advantage of these characteristics, current POMDP solution techniques can be tailored and improved in order to generate the solution at significantly reduced computational expense. The contribution of this thesis is such a tailored algorithm for solving information-gathering problems when represented as a POMDP. This algorithm combines a current POMDP solution algorithm with the structure unique to information-gathering problems to create a more efficient solver. This thesis also presents a detailed case study that highlights the improved algorithm as well as a more quantitative demonstration of the benefit of using a POMDP to solve information-gathering decision problems in engineering design.

In order to facilitate the contributions of this thesis, only a specific subset of information gathering decisions is considered herein. A subset was chosen because it allows the focus of this thesis to remain on the POMDP formalism and the improved algorithm without the need for an extensive description of how to represent all kinds of information-gathering decision problems as POMDPs. Although this thesis considers a specific subset of decision problems within the context of design, the concepts discussed and contributions presented are not unique to this subset, and can likely be expanded to other areas.

This thesis is organized as follows: Section 2 defines the information-gathering decision problem in more detail, as well as specific types of problems considered in this work. Section 3 introduces POMDPs in detail and the currently available solution methods. In Section 4, the POMDP framework is compared to other methods using the case study problem. The improved algorithm is presented in detail in Section 5, followed by a demonstration of the improvement in Section 6, again using the case study problem. This thesis concludes in Section 7 with a discussion and summary.

2. INFORMATION-GATHERING IN ENGINEERING DECISIONS

2.1 Information-gathering Decision Problem Definition

This section provides the definition of the information-gathering decision problem. All decisions are defined within the context of design under uncertainty. This section begins with defining the information-gathering decisions in design, and then moves to a specific subset of this type of decision to be considered in this thesis.

2.1.1 General Information-gathering Decision Problem

In a generic design decision problem under uncertainty, the DM is presented with a number of design alternatives from which they must choose one (there may even be an infinite number of design alternatives such as in the case of a continuous design variable). Each of the design alternatives has any number of uncertain outcomes associated with it, where the outcomes represent the uncertainty in the problem. The likelihood of occurrence of each outcome is described by a probability distribution. Further, each outcome has a defined utility for the occurrence of that outcome via the choice of the associated design alternative. For each design alternative, the expected utility across the range of outcomes can be calculated. The DM must choose one of the available alternatives and then the decision problem terminates.

For the case study scenario presented in Section 1.2, the design alternatives are the different lengths for the components of the two actuation concepts. In other words, each design alternative is a specific value for the length and is associated with either concept A or concept B. The uncertain outcomes for each design alternative are the

possible values of the uncertain material property. For example, consider the simple case where there is only one design length available for each concept (i.e., the DM has already chosen the length for each concept and is only left to pick one of the two concepts). For each concept, the uncertain material property (denote them α and β for concept A and concept B, respectively) can take on any number of values. For simplicity, here the values are considered to be only integers, where $1 \leq \alpha \leq N$ and $1 \leq \beta \leq M$. This example is shown using a decision tree in Figure 4. The square represents the decision that the DM must make (choose one of the two design concepts) and the circles represent a chance event over the possible values of the uncertain material property. The utility of each design is defined as a function of the value of the uncertain property. A more detailed description of decision tree analysis is presented in Section 2.3.

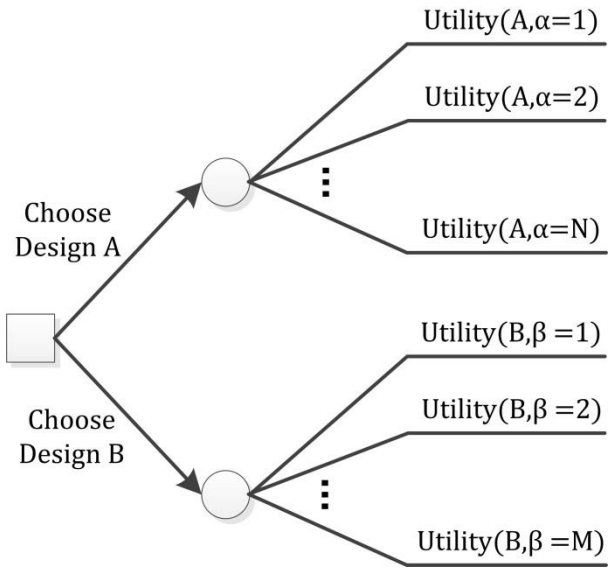

Figure 4. Example decision tree for design under uncertainty.

The general information-gathering decisions problem is an expansion of a typical decision problem where the DM is presented with an alternative (or multiple different alternatives) to gather information about one of the parameters associated with the design, in addition to the design alternatives associated with the decision. In general, a design alternative can represent a design at any level of development, from a newly introduced technological concept to a fully defined design with final specifications. The additional alternative, called an information-gathering action, uses the gathered information to modify the uncertainty in one or more parameters of the design. The DM then returns to the same decision but with better knowledge about the expected outcome of each of the non-information-gathering decision alternatives. This basic problem structure is shown in Figure 5, where the DM will continue to return to the "Gather information?" decision until they choose not to gather any more information. Once this choice is made, a design alternative is chosen and the decision terminates with the expected outcome dependent upon the DM's uncertainty at the time the design alternative is chosen.

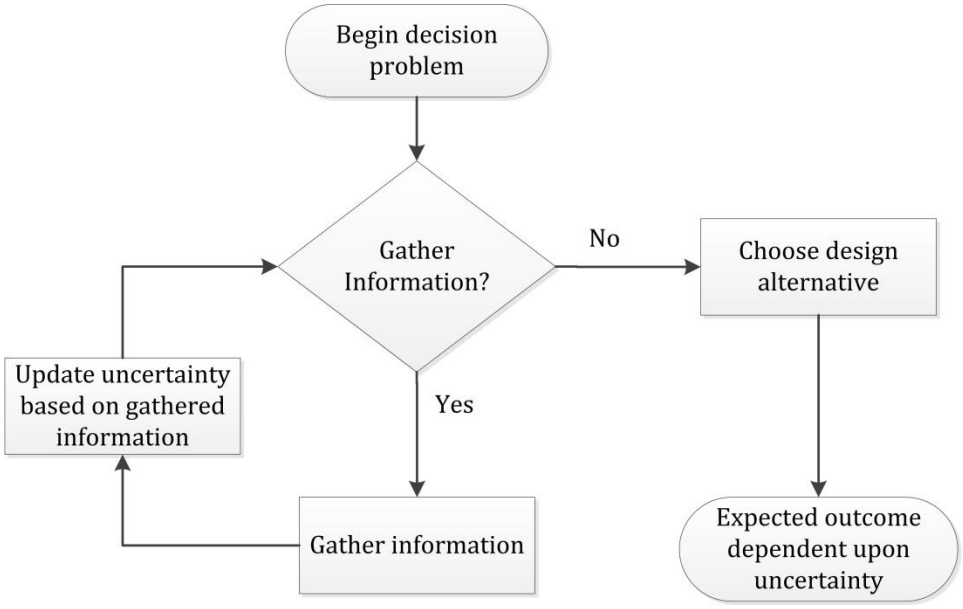

Figure 5. Basic structure of information-gathering decisions.

This yields two types of actions: commitment actions and information-gathering actions. Commitment actions are the actions which result in the DM choosing a design alternative at the current level of uncertainty and terminating the decision problem. On the other hand, information-gathering actions allow the DM to gather information at a specific cost that will likely reduce the uncertainty in one or more parameters of the design. The reduction in uncertainty of the parameters also reduces the uncertainty in the outcome of one or more commitment actions. The DM is then better able to choose from the available design alternatives in order to maximize the expected utility of the design.

Information-gathering decision problems possess an important characteristic: the problem does not terminate until the DM chooses a design alternative (commitment action). If the DM chooses an information-gathering alternative, this only delays

choosing a design. After receiving the gathered information, the DM returns to a similar decision in which they must choose a design alterative or another information-gathering alternative (possibly even the same action that they previously chose). Consider the case study design scenario. If the DM chooses to execute the mechanical testing option for α, they will receive information from the test that will update their uncertainty about α. However, once this is completed, the DM is again presented with the option to execute either of the two mechanical testing options or choose one of the design alternatives. The DM may choose to gather information any number of times before eventually choosing a design alternative.

In the general case, the DM does not return to the exact same decision because certain aspects of the problem could have changed. For example, it is possible that the information-gathering action takes a great deal of time, and one of the design alternatives is only available to the DM if chosen now. In this case, the subsequent decision after gathering information would lack this design alternative. Despite this, any subsequent decision possesses the same general structure as the original decision: choose a design alternative or choose to gather more information.

At the most basic level, the information-gathering decision problem allows the DM to choose a design alternative, or gather information any number of times and then choose a design alternative. However, the DM should only choose to gather information if it is expected to improve the expected utility of choosing a design alternative. If gathering information is not expected to improve the expected utility, the DM should choose from the available design alternatives at the current uncertainty. Because of this,

the DM should determine whether or not it is valuable to gather information at the onset of the decision problem and also after any new information has been gathered. Figure 6 illustrates this decision process including the appropriate criterion for gathering information.

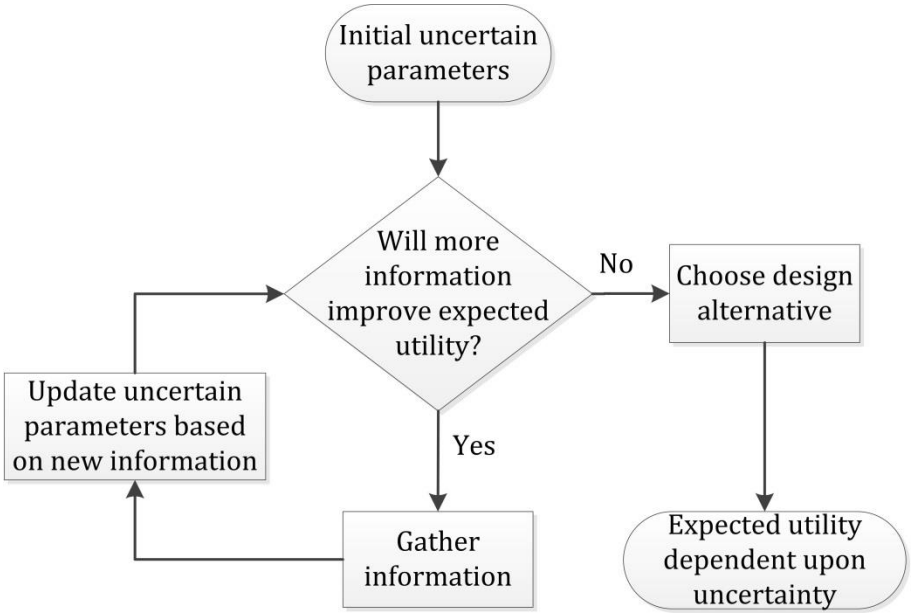

Figure 6. Repeating structure of information-gathering decision problem.

This figure illustrates the nature of this type of problem. Because in general no information-gathering action is perfect, it can be reasonable for a DM to choose to gather information several times in order to further reduce the uncertainty. The DM may choose to pursue several different information-gathering actions or possibly the same action multiple times, which is evident in many engineering scenarios. For example, in engineering modeling and simulation, an engineer often has to choose whether or not to

account for certain physical phenomena (e.g., gravity, heat transfer, nonlinear behavior, etc.). Particularly in design, engineering models are heavily relied upon to predict performance in order to choose a design that will best meet expectations. A model that takes into account more phenomena will be more accurate but likely will be more costly to produce and execute. Thus, an engineer must often consider whether the cost of an improved model is worth the information that will be gained from its execution. This dilemma is also prevalent in material characterization and technology development where engineers must decide when further experimental testing and development, which can take weeks or months, is no longer needed and the material or technology is ready for implementation.

2.1.2 Reduction to Specific Decision Problem

The information-gathering decision problem defined in the previous subsection covers a broad range of decisions in design under uncertainty. For the purposes of this work, a specific subset of this problem type was chosen for two reasons: to better demonstrate the contributions herein and to account for a specific limitation of the POMDP framework. The POMDP framework (explained in detail in the next section) can only accommodate risk-neutral utility functions because it requires the summation of utilities over the duration of the decision problem. Because of this limitation, this thesis will use value measured in dollars in the place of utility. This simplifies the problem representation by not requiring an extra conversion from dollars to utility. All other problem reductions, however, do not represent limitations of this work but were chosen

as a matter of convenience; the contributions presented can easily be expanded to decision problems outside of this subset (save for the risk-neutral requirement).

In this subset, the DM can be presented with any number of decision alternatives and information-gathering actions. However, after choosing an information-gathering action, the DM returns to the structurally identical decision where none of the available alternatives have changed; only the uncertainty has been changed. This requires that the characteristics of every individual information-gathering action (accuracy and cost) remain the same each time it is chosen. There are several types of information-gathering actions that do not fit this requirement, such as deterministic modeling where a model can be executed multiple times, but the answer will be the same after each execution. In this scenario, it would never make sense to execute the model more than once with the same input parameters. Thus this information-gathering action should be removed for subsequent decisions. The contributions in this thesis can be expanded to handle such cases, but the current work focuses on a more basic information-gathering problem definition in order to better highlight the algorithmic contribution.

In this problem subset, the uncertain parameters considered are restricted to those that are independent of the design alternatives. An example of such a parameter is a material property; the material property is independent of the design chosen, but the numeric value of the property affects the outcome, or value, of choosing each design. In other words, this subset does not consider problems where the uncertain parameter is a direct function or characteristic of each design. This assumes that the DM knows the outcome associated with choosing each design with certainty if they know the values of

the uncertain parameters with certainty. In reality, even if the values of the uncertain parameters are known with certainty, it is often possible that the DM is uncertain in the value of choosing each design alternative. While this is not directly considered in this subset of problems, the DM can take the expectation across the uncertainty in the values for each design to get a single value. This expectation can then be used as the certain value of each design, thereby allowing the decision to fit into the subset of information-gathering decisions considered here.

The DM may be uncertain about any number of different parameters in the decision problem, and each uncertain parameter may take on any number of values. In general, each uncertain parameter may be discrete (e.g., quality grade) or continuous (e.g., material property) in nature. For this specific problem, the uncertainty must be able to be modeled as a probabilistic distribution over the uncertain parameter space. The probability distribution mathematically represents the likelihood, or in other words the DM's belief, that the uncertain parameter is truly equal to any of the possible values in the parameter space. In the case of a discrete uncertain parameter, the DM's belief is represented by a probability mass function:

$$b(z) = \Pr(Z = z) \quad \text{for all } z \in \mathcal{Z}, \tag{1}$$

$$\sum_{z \in \mathcal{Z}} b(z) = 1, \tag{2}$$

where Z is the uncertain parameter which can take any value $z \in \mathcal{Z}$, and $b(z)$ is the belief of the DM that uncertain parameter $Z = z$. For a continuous uncertain parameter,

$b(z)$ is now a continuous function (probability density function) where $\mathcal{Z}$ is now a region of the measurable space $\mathbb{R}$:

$$\int_a^b b(z)dz = \Pr(a \leq Z \leq b) \quad \text{for all } a, b \in \mathcal{Z}, \tag{3}$$

$$\int_{-\infty}^{\infty} b(z)dz = 1. \tag{4}$$

In the case of more than one uncertain parameter, there must be a belief for each uncertain parameter, where the uncertain parameters are denoted by Z_1, Z_2, etc. As mentioned previously, for this specific problem, the DM knows the value of choosing any design if they know the uncertain parameter with certainty. As such, the value of any design chosen by the DM is a function of both the design itself and the uncertain parameter. For this work, a value-driven design approach is taken to computing the value of any design [55], which has been applied to many engineering cases [56-58]. Under value-driven design, there are no attribute requirements presented to the designer (e.g., maximum mass, minimum power, etc.). Instead, any combination of attributes is directly mapped to a scalar value, where again the designer seeks to maximize value. This mapping is achieved through a design value function $v(Y)$, which takes as input the vector of attribute values $Y \subset \mathbb{R}^n$ and outputs the associated value v and is specific to each design problem. In addition, the attribute values are a function of both the design alternative x and the uncertain parameter through the appropriate engineering modeling and analysis: $Y = f(x, z)$. The fidelity of this analysis can vary based on the specific design scenario, but some form of analysis is necessary for the engineer to predict how a

design will perform. The overall design value function then becomes a function of the design alternative x and the uncertain variable value z:

$$v = f(x, z) \qquad \text{for all } x \in X \text{ and } z \in \mathcal{Z}. \tag{5}$$

where X is the set of all available design alternatives. Using this design value function under uncertainty, the expected value of any design $x \in X$ is calculated as follows for a discrete uncertain parameter and continuous uncertain parameter, respectively:

$$E_z[v(x, z)] = \sum_{z \in \mathcal{Z}} b(z)v(x, z), \tag{6}$$

$$E_z[v(x, z)] = \int_{\mathcal{Z}} b(z)v(x, z)dz, \tag{7}$$

where the DM chooses the design with the highest expected value x^*:

$$x^* = \max_{x \in X} E_z[v(x, z)]. \tag{8}$$

This work also takes a project oriented perspective of the design process, as opposed to design oriented; the goal is to maximize the net present value of the project as a whole, not simply the value of the final chosen design. The importance of such a perspective is demonstrated in the work of Thompson and Paredis [23], as discussed in Section 1.1. By considering project value, both the costs incurred during the design process and the value of the final design are directly accounted for when making decisions. This allows a DM to make decisions that maximize the overall value of the design project, not just the value of the design that is chosen.

In order to model this perspective, a project value function v_p must be defined to take into account both the value of the design as well as the associated costs of development C in order to calculate the net present value:

$$v_p = v - C, \tag{9}$$

where C may be divided up into as many categories as necessary to incorporate multiple different cost sources, both fixed and variable in nature. Thus, the objective of the DM is to maximize the expected value of v_p. The expected value of v_p, however, depends on the specific beliefs of the DM over the uncertain parameter Z. Because the information-gathering actions modify the DM's belief, maximizing the expected value of v_p requires predicting how the information-gathering action is expected to modify this belief. This is particularly difficult because after each time the DM chooses to gather information, they are again presented with the choice of gathering additional information. This demonstrates the value of modeling the sequential information-gathering problem explicitly due to the complicated tradeoff between the value of the design and the cost of gathering information.

2.2 Explicit Modeling of Sequential Information-Gathering Decisions

In order to perform the explicit modeling of the sequential information-gathering decision presented in Section 2.1, the most common method is decision tree analysis, which is a visual representation of the decision problem [17, 20]. While there are other modeling methods available such as neural networks [59] and influence diagrams [17],

the decision tree representation is the most common due to its clear visual structure. A decision tree is a collection of nodes and lines similar in shape to a tree, which represent a decision problem. The nodes can be either decision nodes, represented by squares, or chances nodes, represented by circles. The lines represent decision alternatives available to the DM or possible outcomes of a chance event, depending on the node from which they originate. A probability is associated with all outcomes of the chance nodes and is often displayed next to the appropriate line. Each branch of the tree terminates at a leaf with a decision alternative outcome, where a value is specified for reaching this leaf of the decision tree. Values, which can be either positive or negative, can also be specified at each decision alternative along the branches. Thus, the total value for each leaf is the sum of all values along the path from the initial decision, or root, to that leaf.

A decision tree for a sample information-gathering decision problem is presented in Figure 7. For simplicity, this problem considers the case where the DM has only one type of information-gathering action available and can only pursue this action once before choosing a design. It is a basic expansion of this tree to incorporate multiple different information-gathering actions as well as the ability to purse each action more than once.

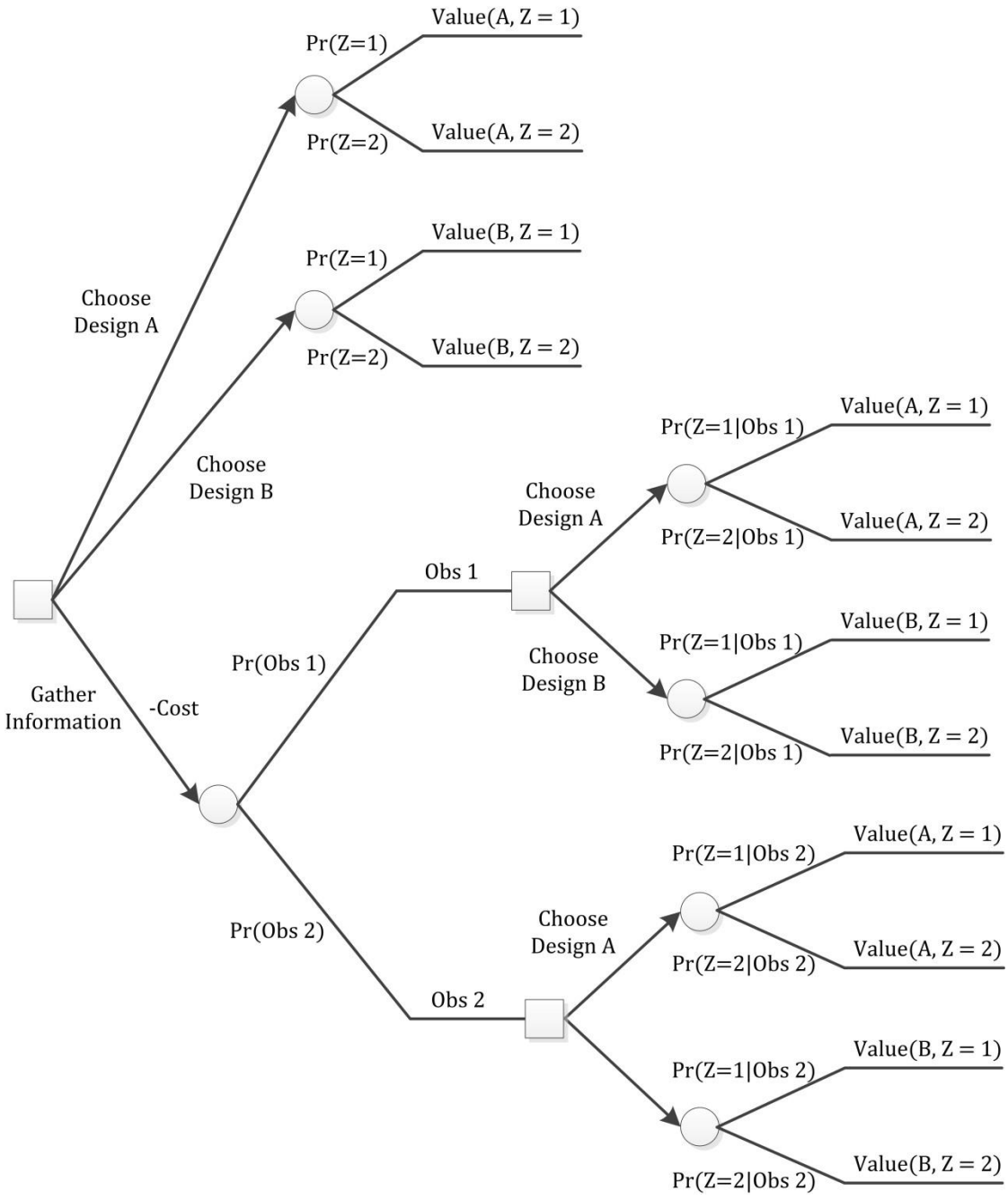

Figure 7. Decision tree for example information-gathering decision problem.

In this problem, there are two design alternatives, A and B (similar to the case study), but only a single uncertain parameter Z which can take on a value of 1 or 2

($|\mathcal{Z}| = 2$) . The numeric value of the uncertain parameter affects the value of choosing designs A or B at each leaf. Because we consider imperfect sources of information in this work, there are multiple possible outcomes from gathering information. Each possible information gathering outcome is referred to as an observation. For this example, there are only two possible observations, representing two different possible sets of gathered information (note that the number of observations was chosen for simplicity and is not necessarily related to the number of values of the uncertain parameter). The information-gathering action has a cost associated with it that is represented with a negative value to indicate that it reduces the final value of all leaves on this branch.

All of the necessary probabilities for the outcomes of the chance nodes are presented in Figure 7 where the probability of an outcome x is denoted by $\Pr(x)$. In the case of the information-gathering action, the observation received (i.e., the observation that the DM receives from the information-gathering action) affects the probability of each value of the uncertain parameter Z. This updated probability, which is a function of the specific observation received, is denoted by $\Pr(Z = z|Obs)$. This is incorporated in the probabilities of all outcomes of the chance nodes on subsequent branches. The values of the outcomes, however, are all unaffected by the observation received because the outcome value depends only on the actual numeric value of Z and the design alternative chosen.

In order to determine the best course of action, the DM begins at the leaves of the tree and works his way back to the root. This is often referred to as "rolling up" the

decision tree, or more formally, backward induction. When the DM encounters a chance node, they must calculate the expected value of all the possible outcomes at that node. The expected value is based on the probability as well as the value of each outcome. For example, take the chance node at the top of the decision tree in Figure 7 that is a result of choosing Design A without gathering information, which is shown again in Figure 8. The expected value of this chance node is calculated as follows, where $\mathrm{Value}(x, Z = z)$ is denoted by $\mathrm{V}(x, z)$, $\Pr(Z = z)$ by $\Pr(z)$, and $\Pr(Z = z|\mathrm{Obs}\ o)$ by $\Pr(z|o)$:

$$Ev = \Pr(1)\mathrm{V}(A, 1) + \Pr(2)\mathrm{V}(A, 2). \quad (10)$$

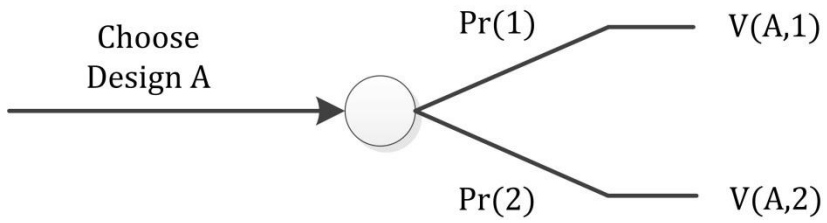

Figure 8. Chance node after choice of Design A without gathering information.

When the DM encounters a decision node, he chooses the alternative that has the highest expected value. Now consider the portion of the decision tree that is a result of receiving Observation 1 from the information-gathering action, shown in Figure 9. The expected value of this decision node is calculated as follows:

$$Ev = \max\{\Pr(1|1)\mathrm{V}(A, 1) + \Pr(2|1)\mathrm{V}(A, 2), \Pr(1|1)\mathrm{V}(B, 1) + \Pr(2|1)\mathrm{V}(B, 2)\}. \quad (11)$$

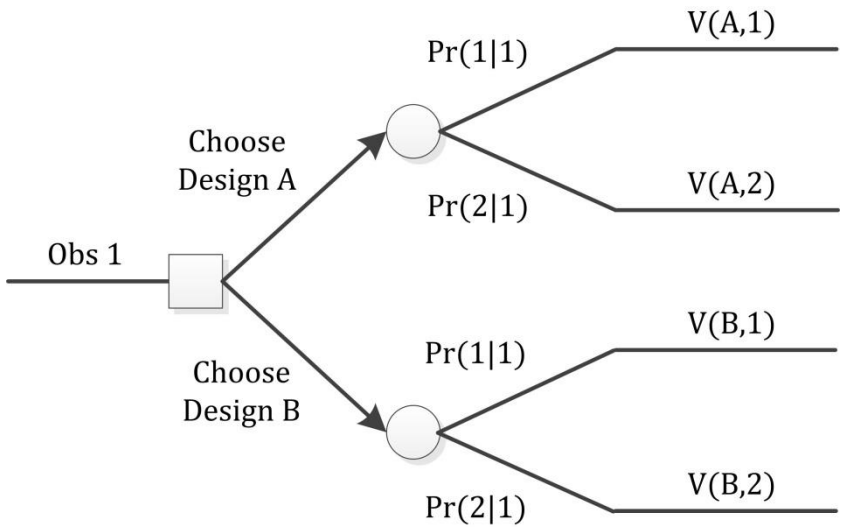

Figure 9. Decision node after receiving Observation 1 from gathering information.

While moving back toward the root, any decision alternatives that carry an associated value, such as the cost of the information-gathering action, are added to the value of the respective branch. Once the DM reaches the root, they have solved the decision tree for the best available decision alternative at each decision node in the tree.

By using a decision tree to visualize and solve an information-gathering decision problem, the DM can explicitly consider the effects of potential future decisions on the current decision. Further, the full solution to the decision tree includes the best decision alternative not only at the root node, but also the best alternatives at any subsequent nodes in the tree. In the information-gathering problem, if the best alternative at the root node is to gather information, the DM already knows the best alternative for the next decision depending on the observation received. As such, the sequential nature of the decision process is fully captured within the decision tree representation.

Although it demonstrates the usefulness of decision tree analysis, the previous decision problem is a very basic example of an information-gathering problem. Not only is the decision tree truncated such that the information-gathering action can only be pursued once, but it also contains very few commitment actions (designs A and B), values of the uncertain parameter, and possible observations received from the information-gathering action. It can easily be seen, by looking at the decision tree, that the size of the decision tree for information-gathering problems grows rapidly with the commitment actions, uncertain parameters, observations, and most importantly, the number of times the information-gathering action can be repeated. For example, the current decision tree has only 12 leaves. By simply adding the ability for the DM to gather information a second time, the number of leaves increases to 28. By the time the decision tree increases to allow for gathering information five times, the number of leaves jumps to 252. This expansion is even further pronounced on a larger problem with more actions, uncertain parameters and observations. Because of this rapid expansion of the size of the decision tree, calculating the solution to large, complicated problems using decision trees quickly becomes impractical [23].

Note that the focus of the work presented in this thesis is not on problems where the DM would anticipate gathering information a large number of times. As mentioned in Section 1, the fields of optimal stopping and optimal sample size are well established. The intent of this demonstration of the increase in decision tree size is only used to convey the rapid increase in size even when gathering information a few times. This

effect increases exponentially under the presence of multiple information-gathering actions and multiple uncertain parameters.

2.3 Repeating Structure of Sequential Information-Gathering Decisions

While the example decision tree presented in the previous section demonstrates the impracticality of solving large problems, it also demonstrates a useful feature of information-gathering problems: the tree has a repeating structure. At every decision node, the commitment actions lead to a chance node over the outcome values. The information-gathering action(s) lead to another decision node (via a chance node) with the same structure as the previous decision node, with only changes to the probabilities and values at the leaves. There is a structural difference if the DM is able to gather information only a finite number of times, in which case the final decision node simply will have no information-gathering action available, or if the characteristics of the information-gathering action change with each evaluation. For the general information-gathering problem, the DM could technically choose to gather information without end, causing the full decision tree to have infinite length. However, the repeating structure of information-gathering problems can be used to define a more basic decision tree that drastically simplifies the problem representation. This decision tree with repeating structure is presented in Figure 10 for the general information-gathering decision problem.

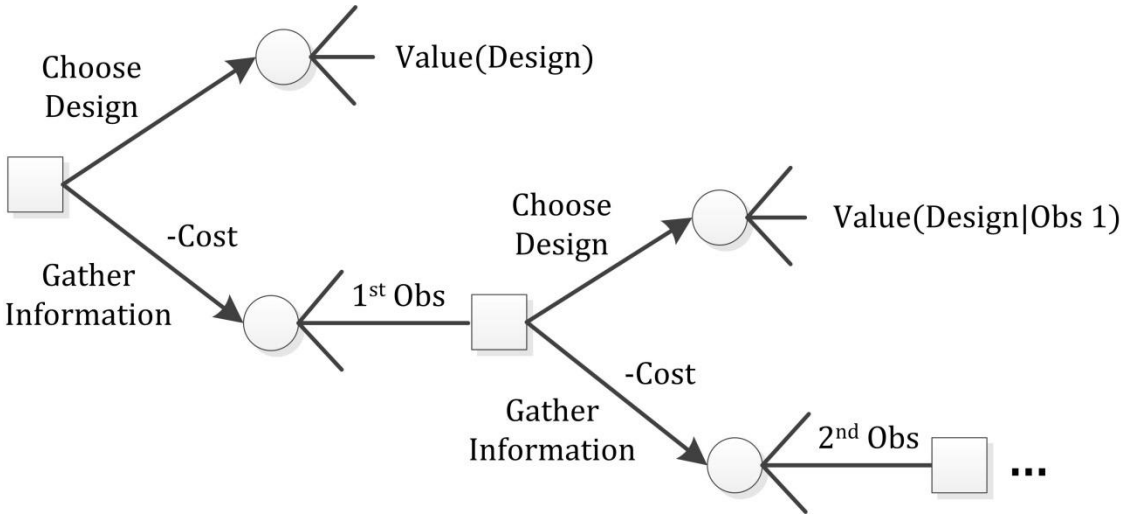

Figure 10. Decision tree with repeating structure for the general information-gathering decision problem.

This representation defines the full, infinite decision tree in a much more succinct and tractable form. The "Choose Design" alternative represents the set of all available design alternatives, allowing for any number of alternatives. Similarly, the chance nodes for the information-gathering actions and the chance nodes over the outcome values allow for any number of observations and uncertain parameter values respectively. Here, "1st Obs" and "2nd Obs" denote the observations received after the first and second decisions to gather information respectively. This decision tree clearly demonstrates the recursive nature of the information-gathering decision problem: after gathering information, the DM moves to a new decision node with the exact same alternatives as the previous decision. For decision tree analysis however, this recursion presents a benefit only in the representation of the decision problem; it does not reduce the complexity of rolling up the tree to generate the solution. However, this recursion can be

used to map the information-gathering decision problem into a recursive decision making formalism.

One such recursive decision making formalism is the partially observable Markov decision process [60]. In a POMDP, the DM occupies one of many possible states at any given time. At each time step, the DM takes an action and the decision process transitions to a new state (including the possibility of transitioning back to the same state) based on a probability distribution. With each action, the DM receives a reward which is based on the current state, the action taken, and the state that the process transitions to. However, during this process, the DM may not perfectly be able to observe what state the process is in. He holds beliefs about the current state, but generally does not know the state with certainty. With each action taken, the DM receives an observation about the current state. While the reliability of this observation depends on the specific problem, the observation affects the DM's beliefs about the current state. The objective of the DM in a POMDP is to choose the action at each step of the decision process that, based on the beliefs about the current state, maximizes the expected reward at the current step and all future steps.

The partial observability inherent to the POMDP formalism allows information-gathering decision problems to be represented explicitly; after each information-gathering action, the DM receives an observation that updates the beliefs over the parameter(s) of interest. The DM then uses the updated belief to decide what to do at the next step. Furthermore, this formalism can take advantage of the recursive nature of information-gathering decision problems when generating the solution, unlike decision

tree analysis. By taking advantage of the recursion, the POMDP formalism makes many complicated decision problems practically solvable that are impractical when using a decision tree.

3. PARTIALLY OBSERVBABLE MARKOV DECISION PROCESSES (POMDP)

3.1 Introduction to POMDPs

In order to describe a POMDP, it is useful to begin with the description of a Markov decision process (MPD), because a POMDP is an extension of the MDP formalism for recursive decision making [60, 61].

3.1.1 MDP Description

In the MDP formalism for modeling decision-making, there are four major elements: states, actions, transition functions and rewards [61]. For a given problem, there are a set of states, where the decision process occupies one of these states at any given time step. The DM has a number of available actions at each step, which may change the state that the decision process occupies. The transition between states, based on the DM's action, is stochastic and modeled by a transition function that captures the probability of moving from one state to another as a function of each action. Every action taken by the DM results in a reward, that is based on the current state of the process and the specific action chosen. Any MDP can be formally defined by the following parameters:

- S is the finite set of states of the world, or problem, where a single state is $s \epsilon S$. Only one state can be occupied at each step.
- A is the finite set of actions the DM can take, where a single action is $a \epsilon A$. The DM can only take action at each step.
- $T: S \times A \times S \rightarrow [0,1]$ is the stochastic transition function that takes as an input the current state, action chosen by the DM, and a new state of the process and

returns a probability for that transition. In other words, this is the probability that the decision process moves to that new state after the associated combination of action and current state. The probability of moving to each new state s' is denoted by $T(s, a, s')$ where s is the current state and a is the action chosen.

- $R: S \times A \to \mathbb{R}$ is the reward function that also takes as input the current state and action chosen and returns the expected value of the state-action pair. This expected reward is denoted by $R(s, a)$. In general, the reward can also depend on the new state s', giving the form $R(s, a, s')$; however, this can be translated to $R(s, a)$ by simply taking the expectation over the new states.

As a simple example of a Markov Decision Process, Figure 11 presents a process with only two states, $s1$ and $s2$. At each state, the DM has the same two actions available: $a1$ and $a2$. The transition function T and reward function R for every combination of current state, action, and new state is shown along the corresponding paths. As noted above, the reward function can be reduced to only a function of the current state and action by calculating the expectation over the new states. These values are shown in the bottom right corner of Figure 11. Additionally, the transition function is presented in a slightly different form in order to place the emphasis on the action taken: $\boldsymbol{a}: T(s, s') = T(s, a, s')$.

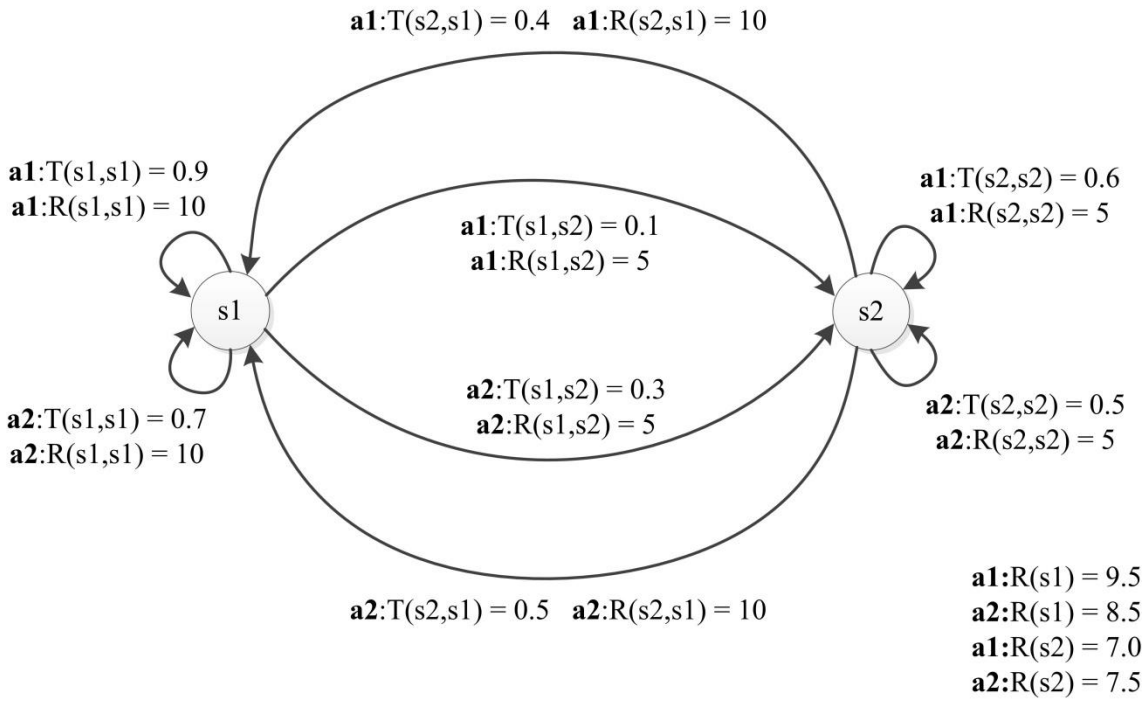

Figure 11. Sample MDP with two states and two actions.

The DM seeks to maximize the rewards, or values, over the lifetime of the decision process by choosing actions optimally at each step (one action at each step), thereby explicitly considering the sequential nature of the process. By analyzing these parameters and solving the MDP, the following can be obtained:

- $\pi: S \rightarrow A$ is a policy which takes the current state as input and returns to the DM the best action to take.
- $V(s)$ is the value function that determines the expected value of taking the best available action based on the current state s.

In the case of the example in Figure 11, the policy π is a function of the two states $s1$ and $s2$. Depending on the current state, the policy outputs the optimal action for the DM to take that maximizes the expected rewards over the lifetime of the decision process. Note that this policy (and the MDP itself) possesses the Markov property: it is memoryless in that it does not take into account or keep track of the past actions taken [61]. The value function V gives the expected reward over the lifetime of the process corresponding to the optimal action from the policy. While in general the policy is not easily found, the policy for Figure 11 is rather simple: $\pi(s1) = a1,\ \pi(s2) = a2$. This is due to the fact that it is clearly more valuable to move to or stay in state $s1$, as is evidenced by the fact that all rewards moving to $s1$ are double that of $s2$. Thus, a check of the transition probabilities reveals that, when the current state is $s1$, the process is more likely to move to $s2$ with action $a1$: $T(s1, a1, s1) > T(s1, a2, s2)$. Similarly for the current state being $s2$, action $a2$ is preferred: $T(s2, a2, s1) > T(s2, a1, s2)$. Again, for most MDPs, the policy is not found intuitively and is much more complicated.

3.1.2 POMDP Description

The MDP formalism clearly accounts for the stochastic nature of sequential decisions, but it assumes that the state of the process can be known with certainty by the DM. For many processes, this is not possible because the DM may not be able to observe the true state. A partially observable Markov decision process relaxes this assumption such that the DM holds *beliefs* about which state the process may be in [48, 60]. These beliefs form a probability distribution over the states and must sum to one:

the belief in a particular state is the probability of the process occupying that state. When the DM takes an action at each step, they receive an observation that is based on the true state, and may cause the beliefs over the states to change. This observation does not necessarily directly reveal the true state, because the framework allows for the same observation to come from multiple combinations of states and actions. Similar to the transition function, the observation function defines the probability of receiving an observation as a function of the state and action. In addition to all of the parameters required for an MDP, the following additional parameters must be defined for a POMDP:

- Ω is a set of all possible observations that the DM can receive about the state of the world, where a single observation is $o \epsilon \Omega$.
- $O: S \times A \times \Omega \rightarrow [0,1]$ is the stochastic observation function that takes as input the action chosen, the new state of the world as a result of this action, and a specific observation. This function returns the probability that the DM receives that observation based on the associated combination of action and new state. The probability of receiving each observation o is based on the action taken a and new state of the POMDP s' and is denoted by $O(a, s', o)$. In this case, the observation is dependent upon the new state of the POMDP, not the current state as with the transition function.
- B is the set of all possible belief distributions over the states of the world. Here, $b \epsilon B$ comprises a single belief distribution and the each component $b(s)$ denotes the probability, or the DM's belief, of the world occupying state s.

The objective of the DM is to maximize the expected future rewards by choosing the most valuable action at each step of the decision process according to the optimal policy. This is the same objective for the DM as with the MDP; however, this objective is more difficult to achieve because the DM does not know the state of the process when choosing actions. As a result, the optimal policy for the DM is now a function of the beliefs instead of the states:

- $\pi: B \rightarrow A$ is a policy which takes the current belief as input and returns to the DM the best action to take.
- $V(b)$ is the value function that determines the expected value of taking the best available action based on the belief b.

Using the policy $\pi(b)$, the DM can learn the best action solely as a function of the a priori belief about the current state of the problem. When the DM takes an action, an observation is received and the belief must then be updated to incorporate the observation. The belief updating is accomplished through Bayes' Rule at each observation where $b'(s')$ is the updated belief in the new state of the problem s', which is the state of the problem after the action has been taken:

$$b'(s') = \Pr(s'|o,a,b) = \frac{\Pr(o|s',a,b)\,\mathrm{p}(s'|a,b)}{\Pr(o|a,b)}$$

$$= \frac{\Pr(o|s',a)\sum_{s\epsilon S}\Pr(s'|a,b,s)\Pr(s|a,b)}{\Pr(o|a,b)}$$

$$= \frac{O(a, s', o) \sum_{s \epsilon S} \mathrm{T(s, a, s')b(s)}}{\Pr(o|a, b)}. \quad (12)$$

Here, the denominator is a normalization factor that ensures that the sum of the elements of b' is one:

$$\Pr(o|a, b) = \sum_{s' \epsilon S} \left(O(a, s', o) \sum_{s \epsilon S} T(s, a, s')\, b(s) \right). \quad (13)$$

One of the key benefits of the policy is that it is valid over the entire belief space; no matter what the DM's beliefs are, the policy will give the best action to take. Thus, when the DM chooses an action that results in updating the beliefs, such as gathering information, the same policy is valid to determine the next action even with the new belief.

In the example process presented previously, the current state is now hidden from the DM such that they only have a belief about whether the process is in state $s1$ or $s2$. For example, the DM may be completely unsure about the current state such that the belief is uniform over both states: $b(s1) = b(s2) = 0.5$. Because of this, the DM must make his decision on which action to take solely based on the beliefs. After each action taken, the DM receives an observation that is related to the action chosen and the new state (which becomes the current state for the next step) of the process through the stochastic observation function. Using Bayes' Rule, the DM's belief over the two states is updated to reflect the information that the observation provides. This updated belief then becomes the belief used to choose the best action at the next step. In order to solve

this problem, both the observation set and the observation function must be defined. Due to the added complexity from the partial observability, this type of problem is more difficult to solve than the corresponding MDP.

3.2 POMDP Solution Methods

The solution to a POMDP is the optimal policy for the DM to follow. In order to generate the optimal policy, there are multiple methods available for general POMDPs, including multiple freely available solvers for download on the internet. However, a solution to a general POMDP may require significant computational expense to attain, because there is no limit to the number of states, actions and observations that can be included in a process. The solution to a general POMDP has proven to be PSPACE-complete for a finite-horizon (i.e., the problem terminates after a finite number of steps) [62], and undecidable for the infinite-horizon case [63]. A full review of POMDP solution techniques is beyond the scope of this thesis, but a brief review is presented here.

In order to understand and compare POMDP solution methods, it is first helpful to learn about MDP solution methods because most POMDP methods are modifications and/or extensions of classic MDP solution methods. Furthermore, this discussion will be limited to infinite-horizon models using the discounted total reward criterion. A finite-horizon POMDP is a decision problem where the number of time steps, or decisions, the DM may make is limited to a predetermined, finite number; the decision problem terminates when this limit is reached. By comparison, the infinite-horizon POMDP does

not place any limit on the number of steps, allowing the problem to theoretically continue without end. In order to address the potentially infinite length of the infinite-horizon problem, the discounted reward criterion defines a discount factor, $\gamma \epsilon [0,1)$, which reduces the rewards of future actions: after the decision at each time step, the value of all rewards at the next step are scaled by the discount factor. The present value of rewards obtained at future time steps is discounted by γ^n, where n is the time step. The POMDP representation requires the presence of a discount factor for the infinite-horizon in order to guarantee that the DM cannot gain infinite rewards. However, for many information-gathering problems, a discount factor is not needed because many design projects do not last long enough for a discount factor to be relevant. Thus, the requirement of a discount factor in the POMDP representation can be an issue when modeling information-gathering decision problems as POMDPs. This is addressed in more detail in Section 5, where the improved algorithm addresses this issue.

The methods discussed in this section can usually be applied to finite-horizon problems and are generally less computationally expensive due to a finite number of time steps. In an infinite-horizon problem, there exists an optimal stationary policy for both an MDP and POMDP (i.e. a policy that is independent of the time step). This makes the problem much more approachable, as a policy does not need to be defined differently at every time step [54]. While several other algorithms exist, those discussed are the most recognized in the literature [61].

3.2.1 Value Iteration

Value iteration is the most widely used algorithm for solving MDPs. This method relies on the Bellman equation, or the Bellman update [54]:

$$V_{i+1}^{*}(s) = \max_{a \in A} R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') V_i^{*}(s'). \tag{14}$$

Here, $V_i^*(s)$ is a finite-horizon value function and is the value of occupying state s and taking the best available action, where i is the number of steps left to problem termination and as $i \to \infty$ (i.e. the infinite-horizon case), $V_{i+1}^* = V_i^*$. This update can be thought of as taking a finite-horizon value function V_i^* and adding an additional step at the beginning to generate the value function for the finite-horizon case with $i + 1$ steps V_{i+1}^*. By initiating V_0^* to any value function, the Bellman update can be iteratively applied until the difference between V_{i+1}^* and V_i^* is sufficiently small. This process of value iteration is known to converge toward a single unique fixed point for each state. In order to develop the optimal policy, after each iteration the corresponding action that maximizes the Bellman update is recorded as the appropriate action in the optimal policy. The resulting policy after the last iteration is the optimal policy and is guaranteed to converge. In fact, once the difference between successive iterations (defined by the average difference between the value of all belief points) is less than some value ε, the difference between the optimal policy and the true solution is guaranteed to be less than $2\varepsilon\gamma/(1-\gamma)$ [54].

Value iteration for POMDPs is based on the value iteration method for MDPs, which relies on the fact that a POMDP can be represented as a continuous belief-state MDP [52]; instead of discrete states, the MDP has a continuous range of states, where each state represents a possible belief over the states of the POMDP. Generally, POMDP value iteration works in the same way as for MDPs, but must account for the infinite number of states. The Bellman update then becomes the following, where $b^{a,o}$ is the updated belief of b after a particular action and received observation:

$$V_{i+1}^{*}(b) = \max_{a \in A} R(b,a) + \gamma \sum_{b' \in B} T(b,a,b') V_i^{*}(b')$$

$$= \max_{a \in A} R(b,a) + \gamma \sum_{o \in \Omega} \Pr(o|b,a)\, V_i^{*}(b^{a,o}). \quad (15)$$

While the belief space B is infinite, this can be overcome by utilizing the fact that the optimal value function for a POMDP can be approximated arbitrarily closely by the upper envelope of a set of linear functions, known as α vectors [54]. Each vector is associated with a particular action and has one element for every state. As such, the value function $V(b)$ is defined by the set of α vectors, and the expected value is calculated as follows:

$$V(b) = \max_{\alpha \in V} \sum\nolimits_{s \in S} b(s)\alpha(s). \quad (16)$$

The α vectors also comprise the policy π where the best action to take is the action associated with the α vector that maximizes $V(b)$. By beginning the first iteration with

V_0^* consisting of some initial set of α vectors, the number of α vectors grows with each iteration, causing V_i^* to incrementally move closer to the optimal value function.

There are many methods by which the α vectors are updated, with some methods being exact and others approximate. Even for approximate algorithms, this process is extremely expensive because the number of α vectors increases exponentially with each iteration. It is the handling of these α vectors that differentiates many value iteration algorithms as they attempt to reduce the computational complexity. The most basic approach is to enumerate all possible combinations of actions and observations at each iteration. More sophisticated algorithms attempt to reduce the size of this set at each iteration. For example, some algorithms prune the set of α vectors to a dominated set, while others attempt to generate the dominated α vectors directly from the previous iteration's dominated set [48, 52].

3.2.2 Policy Iteration

Policy iteration for MDPs directly updates the policy rather than indirectly finding it via the value function through the Bellman update. First the policy is initialized to some constant policy. Using the Bellman update iteratively until convergence, the value function for the current policy V_{π_i} is computed. Once converged, the policy is updated using the one-step look ahead with the converged value function as follows [52]:

$$\pi_{i+1}(s) = \underset{a \in A}{\operatorname{argmax}} R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') V_{\pi_i}(s'). \tag{17}$$

These steps are repeated until the policy ceases to change. The main difference between policy and value iteration is that value iteration loops directly around the Bellman update (in other words, iterating around the value function) and indirectly creates the policy from the Bellman update. Policy iteration, instead, loops directly around the policy. This method is also guaranteed to converge and under certain conditions can converge faster than value iteration.

Policy iteration becomes much more complex when moving to POMDPs. It first uses a finite state controller to directly represent the policy. A finite state controller is a method of approximately representing a non-stationary policy (a policy dependent on the time step) as a stationary policy, by assuming the controller has a finite set of memory to represent belief states. Some problems can be represented exactly in this way, while others will be close approximations [52]. The policy iteration then proceeds in the same manner as for MDPs, but iterates around and improves the finite state controllers. Iteration terminates when successive controllers are identical.

3.2.3 Point-Based Value Iteration

In general, both policy iteration and value iteration do not scale well to large POMDPs due to the computational complexity of the problems. The Bellman update alone demonstrates the growth in complexity as the number of states and actions increase. Using either of these methods as described is not sufficient for solving large problems; however, a more recent and efficient approach to value iteration has made solving large POMDPs much more realistic. Point-based value iteration (PBVI) is an

approximate value iteration algorithm that restricts the infinite belief space to a finite subset [54]. This finite subset approximates the infinite space and significantly reduces the computational complexity by limiting the exponential growth of the α vectors that represent the value function. Although PBVI provides an approximate solution, it is a much more realistic method of solving large POMDPs. This is essential to using POMDPs to represent information-gathering problems because real-world problems are rarely represented as small POMDPs.

While there are different ways to generate this subset, the general procedure of PBVI is presented here. First, a set of belief points is chosen to represent the infinite belief space. The specific method by which these points are chosen depends on the algorithm, but they are chosen such that each point is a reasonable belief of the DM. Choosing belief points at random or at regular intervals over the belief space may result in beliefs that are highly unlikely for the DM to have. As such, PBVI algorithms use varying methods to generate a representative belief subset. In addition, some algorithms update or expand this subset during each iteration while others work with a constant subset.

With the belief subset chosen, PBVI updates the value function by maintaining only α vectors that are optimal for at least one belief point in the subset. This avoids the need to perform a full Bellman backup of the value function, which would create and maintain several α vectors that are only optimal for beliefs outside of the chosen belief subset. The Bellman backup of a single belief b can be generated through manipulation

of the value function update procedure, where $\text{backup}(V, b)$ is the updated α vector for the belief b [54]:

$$\text{backup}(V, b) = \underset{\alpha_a^b : a \in A, \alpha \in V}{\text{argmax}} \ (b \cdot \alpha_a^b), \tag{18}$$

$$\alpha_a^b = R(s, a) + \gamma \sum_{o \in \Omega} \underset{\alpha^{a,o} : \alpha \in V}{\text{argmax}} (b \cdot \alpha^{a,o}), \tag{19}$$

$$\alpha^{a,o}(s) = \sum_{s' \in S} \alpha(s') \, O(a, s', o) T(s, a, s'). \tag{20}$$

The backup equation twice prunes dominated vectors, thereby generating only one new α vector per belief point. A full backup of all belief points in the belief subset is required to finish a single iteration. Again, each PBVI algorithm is different in generating the full backup from the individual belief point backups. However, they all make use of the fact that the $\alpha^{a,o}$ vectors, referred to as look-ahead α vectors, are independent of b. As such, they need only be calculated once at the beginning of each iteration and cached, saving computational expense. Using this backup method, PBVI proceeds like normal value iteration by establishing an initial value function and iterating until the change in the value function is sufficiently small.

3.2.4 Perseus: Randomized Point-based Value Iteration for POMDPs

Each PBVI algorithm approaches the use of the belief subset and the Bellman backup differently, attempting to reduce computation through various methods. While there are several PBVI algorithms available in the literature, the Perseus solver

developed by Spaan and Vlassis [64] was chosen for this work as the PBVI algorithm to improve upon. The overall Perseus procedure is presented in Figure 12.

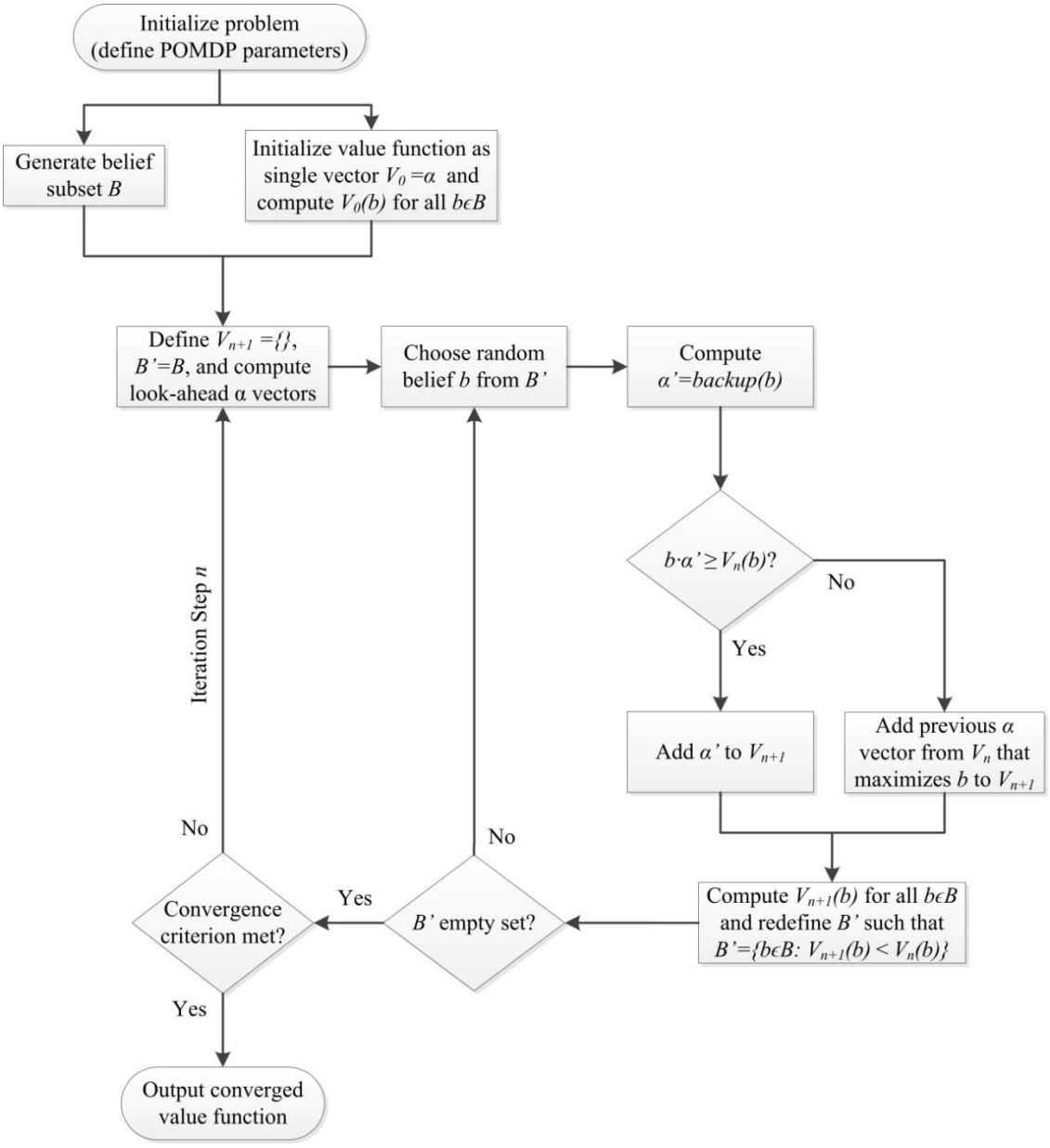

Figure 12. Diagram of Perseus algorithmic steps.

The Perseus algorithm approaches both the selection of the belief subset and Bellman backup procedure in unique ways. For the generation of the belief subset B, Perseus begins with the initial belief of the DM (defined as the uniform distribution if not specified) and selects an action at random. Based on the action, an observation is also received stochastically (based on the observation function) and the belief update procedure, via Bayes' Rule, is used to generate the associated new belief. This procedure is repeated several times where, after each action and observation, the updated belief is added to the belief subset. Once a certain number of actions have been taken, the belief is reset to the initial belief and the process begins again. This is repeated until the belief subset reaches a specified size. By following this method, Perseus ensures that all of the beliefs in B are feasible and realistic beliefs of the DM.

Next, the value function is initialized as a single α vector ($V_0 = \alpha$) and the value of every belief point is calculated by $V_0(b)$ accordingly. For Perseus, and many other PBVI algorithms, the initial value function must always be a lower bound on the converged solution such that each iteration increases the value function toward the true solution [54]. In order to ensure this, Perseus initializes the value function to a single α vector with all of the components equal to $\frac{1}{1-\gamma}\min_{s,a} R(s,a)$, which is equivalent to the present value of receiving the lowest possible reward at every step over the infinite horizon. This is necessary because, in general, no prior knowledge is available about the solution to the POMDP, so the lower bound must be established as the worst possible case.

Perseus then enters the Bellman backup iteration. At the beginning of every iteration, the value function for the current iteration V_{n+1} is initialized as an empty set, and an auxiliary set B' is defined as $B' = B$. With these defined, instead of performing the $backup$ for every belief point in B', Perseus randomly chooses a single belief point $b \epsilon B'$ and performs the $backup(V_n, b)$ generating in a new α vector. The algorithm then checks to see if this new α vector improves the value of b from the previous iteration (i.e., if $b \cdot \alpha \geq V_n(b)$). The case of $b \cdot \alpha = V_n(b)$ is included in the inequality for the case that Perseus finds the true solution for one or more belief points; if the true solution is found for a belief, the value will cease to increase at subsequent iterations. If the new α vector results in an improvement for the value of b, the vector is added to V_{n+1}. If not, the α vector associated with b from the previous iteration is added to V_{n+1}. In either case, the algorithm then checks to see if the added α vector improves the value of any other beliefs in B' by computing $V_{n+1}(b)$ for all $b \in B'$. The belief b, and all other beliefs such that $V_{n+1}(b) \geq V_n(b)$, are then removed from B' so that B' only contains unimproved belief points. Next, a new belief point is then chosen from B' and the backup process is repeated until B' becomes an empty set, thus ending the current iteration. The entire algorithm halts once the convergence criterion ϵ is sufficiently low. While there are several criteria available, for this work the relative change in the sum of the values of all belief points is used as follows:

$$\epsilon = \frac{\sum_{b \epsilon B} V_{n+1}(b)}{\sum_{b \epsilon B} V_n(b)} - 1. \tag{21}$$

This method of performing the Bellman backup reduces the number of $backup(V, b)$ function calls while still ensuring that the values of all belief points improve during every iteration. It does so by checking to see if each new α vector improves any other belief points in B'. For any points that this is true, they are removed from B', thereby reducing the number of points that must be backed up while still improving the value of all belief points during each iteration. Further, by selecting the belief subset such that all points are feasible beliefs of the DM, the algorithm avoids solving the POMDP in areas of the belief space that are not of interest to the DM, thus saving additional computational expense. These aspects of Perseus make it a suitable algorithm for applying the improvements presented later in this work.

3.3 Information-gathering Problem Formulation as POMDP

As previously described, this thesis focuses on a specific subset of information-gathering problems. Because the POMDP formalism can be used to represent many types of decision problems, the specific subset of information-gathering problems considered in this work is also a subset of all POMDPs. In this problem type, the DM is presented with a certain number of design alternatives to choose from, but has uncertainty in a given number of parameters of the problem (see Section 2.1). The DM knows with certainty the value of each design if they know the values of the uncertain parameters. The DM also has one or more information-gathering actions which reduce the uncertainty in one or more of the uncertain parameters for a specific cost (negative value).

As previously described, the DM may be uncertain about any number of different parameters in the decision problem, where each parameter is either discrete or continuous. In the preliminary work by Hsiao and Malak with POMDPs, continuous parameters were not considered [47]. Expanding upon this previous work, this thesis includes the incorporation of continuous parameters, albeit in an approximate manner: the parameter must be discretized into a finite number of values. This discretization is necessary because of the current limitations of the POMDP formalism and solution methods presented in the previous sections. Although the POMDP formalism is capable of handling continuous parameters, current solution methods for such problems are limited to extremely simple problems, require certain limiting assumptions, or utilize a similar discretization procedure [65-67]. As such, the subset of decision problems considered herein requires that there be a finite number of values of each uncertain parameter; all continuous parameters must be discretized. Similarly, the number of design alternatives must be discretized as well in the case of a continuous decision alternative space.

The discretization of a continuous parameter is achieved by choosing a representative and realistic range for the parameter using engineering judgment, and discretizing the range into a finite number of values. The fidelity, or accuracy, of the representation is determined by the number of states used to approximate the continuous range; the more states for a given range, the higher the fidelity. In addition to requiring a finite number of values of the uncertain parameter, this subset also requires that there be a finite number of observations generated by the information-gathering actions. The term

"observation" is used to describe the specific information that is provided to the DM. For example, an observation may be the numeric value of the mass of an object as output by a scale. As such, the same discretization procedure is used when considering an information-gathering action that is capable of producing a continuous range of observations.

With the discretization of continuous parameters defined, this type of problem can be formulated as a POMDP by defining all of the POMDP parameters presented in Section 3.1.1. First, the relevant states must be defined. In the information-gathering problem type, the relevant states are the possible values (combinations) of the uncertain parameter(s). For example, consider the case of two uncertain parameters Z_1 and Z_2, each with 10 possible values. The state space must include every combination of these two parameters ($Z_1 = z_1^j, Z_2 = z_2^k$), for j, k =1:10 giving 10^2=100 combinations where each state is denoted by s_i, where $i = 1{:}\,100$. This represents every possible true state of the world, or true values of uncertain parameters at the time that the DM must make the decision (choose a design alternative or gather information). In addition, there is one extra state called the absorbing state, denoted by Z^*. Once the DM chooses a design alternative, the state of the problem moves to the absorbing state, which represents the end of the problem. This is necessary because the infinite-horizon POMDP is by definition infinite, so the absorbing state provides a method to signify the end of the decision problem. Thus, for the example presented above, there would be 101 total states.

Because every state represents a combination of the possible uncertain parameter values, the DM's beliefs must be converted to this framework. Using the definition of the belief b presented in Section 2.1.2, the DM will hold a belief about each uncertain parameter. For the purposes of this work, these will be called partial beliefs b^* (they are called "partial" in order to highlight the fact that each partial belief describes only a single uncertain parameter and does not contain any information about any other uncertain parameters). Using the example from above with two uncertain parameters, the DM will have a partial belief over the values of parameter Z_1 and Z_2, denoted by $b_1^*(z_1^j)$ for $z_1^j \in Z_1$ and $b_2^*(z_2^k)$ for $z_2^k \in Z_2$ respectively. In order to convert these to one, all-encompassing belief b, which is required for the POMDP framework, the product of the partial beliefs is used as follows:

$$b\left(\left(Z_1 = z_1^j, Z_2 = z_2^k\right)\right) = b_1^*(z_1^j) b_2^*(z_2^k), \tag{22}$$

where it is important to note that this definition of b requires that the uncertain parameters be independent. Although this does represent a limitation of the POMDP formalism, this is a common assumption made in many information-gathering decision problems.

As mentioned in Section 2, in the case of a continuous uncertain parameter, the parameter must be discretized over a feasible region. For many problems, a very fine discretization may be needed such that the number of states becomes very large. Particularly in problems with more than one uncertain parameter, the state space can

rapidly grow with the discretization. This demonstrates the need for a POMDP solution method that can handle large problems.

Now considering the actions, the DM has both commitment actions (design alternatives) x_j, where j is the number of alternatives, and the information-gathering actions t_k, where k is the number of unique information–gathering actions. Thus the number of actions is the sum of j and k. After choosing a commitment action, the DM receives a reward based on the state of the problem and the specific design alternative chosen. Additionally, the state of the problem moves to the absorbing state to mark the end of the problem. For the information-gathering actions, the DM receives a constant negative reward defined separately for each action, and the state of the problem remains the same. Intuitively, this is due to the fact that the information-gathering decision does not change the true state of the uncertain parameters or terminate the problem; it only changes the DM's beliefs about the true value.

The transition function is, in general, based on both the current state and the action chosen by the DM; however, due to the definition of the actions above, the transition function for this problem type is only dependent upon the action. The state stays the same with certainty (i.e., probability of one) if an information-gathering action is chosen and the state moves to the absorbing state with certainty if a commitment action is chosen. Thus, the only non-zero transition probabilities are:

$$T(s,a,s') = \begin{cases} 1 \text{ if } a = x_j \text{ and } s' = Z^* \\ 1 \text{ if } a = t_k \text{ and } s = s' \end{cases}. \tag{23}$$

For the rewards, the information-gathering actions carry a specific cost (negative reward) regardless of the true state of the problem. Thus, $R(s,a) = R(a)$ when $a = t_k$. When a commitment action is chosen, the reward is dependent upon both the state and action. These rewards are completely problem-specific; the rest of the POMDP parameters (observations and observation function) depend completely on the specific scenario for the information-gathering decision problem. While purely a matter of convenience, it will be assumed that there is exactly one observation for every possible value of the uncertain parameter(s). For the example given above, this would result in 20 observations. In many scenarios, the information-gathering actions provide a direct observation about the true state of the uncertain parameter. For example, a DM may be uncertain about the quality grade of a material, but a certain testing method may output this quality grade directly as the observation. As such, the number of possible observations (quality grades) from the test matches the number of possible states (quality grades) of the material. All sample problems in this thesis present problems with this characteristic, but this is only a matter of convenience and is not a limitation of the contributions herein.

4. ENGINEERING CASE STUDY DESCRIPTION AND DEMONSTRATION OF BENEFIT OF POMDP FORMALISM

This section presents the engineering case study in more detail and demonstrates the general benefit of using the POMDP formalism to solve information-gathering decision problems in engineering design. This detailed description will also be used in later sections to demonstrate the benefit of an improved POMDP solution algorithm over Perseus. As mentioned in Section 1.2, the context of this case study is the design of a deployment mechanism for an origami-inspired solar array to be used on a space mission by NASA. The solar array is shown again here in Figure 13 (Copyright 2013 by Daily Herald) for reference.

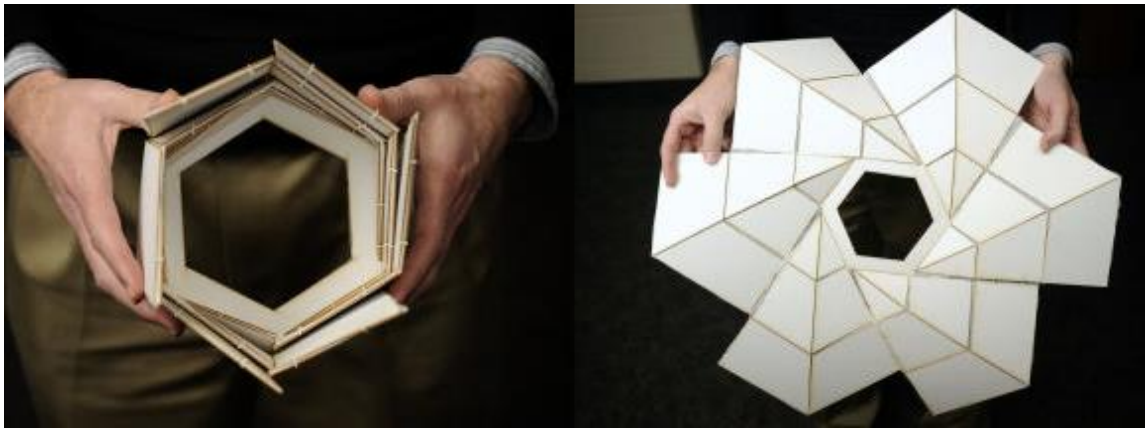

Figure 13. Origami-inspired solar array in both the stowed (left) and open (right) configurations [46].

Following the definition of the design value function in Section 2.1.2, the project value is a function of both the value of the design itself as well as the costs involved in

the design process. While there are generally multiple sources of cost in an engineering project, this case study will focus solely on the costs associated with gathering information. This reduction is appropriate because the only alternatives presented to the DM in this case study are design alternatives and information-gathering alternatives. As such, the only variable cost associated with this study is the cost of gathering information, varying with the number of times information is gathered (and which information source is chosen). Any other costs would be fixed (at least as they are represented in this study), because no decision alternative affects their magnitude. Thus, the additional fixed cost would serve only as a universal offset to the project value and not affect the decision of whether or not to gather information.

Because the cost of gathering information C_t is directly related to the number of times information is gathered and all other costs are neglected, the project value function is defined as follows:

$$v_p = v_d - C_t = v_d - n_{t_1} C_{t_1} - n_{t_2} C_{t_2}, \tag{24}$$

where n_{t_1} and n_{t_2} are the number of times each of the two information actions are executed and C_{t_1} and C_{t_2} are the associated costs of each action. The objective of the DM is to maximize this project value function by choosing one of the information-gathering actions or one of the available design alternatives.

While the engineering scenario for this case study is the design of a deployment mechanism, the focus is on the information-gathering methodology and improved solution algorithm rather than the particular design artifacts. All engineering analysis

conducted is of low fidelity with many assumptions made in order to simplify the design problem representation. This is not needed in order to use the POMDP formalism of IGP, but only serves to keep the focus of this thesis on the information-gathering methodology. As such, this case study is not intended to recommend a design for the origami-inspired solar array, but demonstrate a decision-making methodology for designing such a solar array using a much more detailed engineering representation.

4.1 Case Study Design Scenario

An engineering design firm has been contracted by NASA/JPL to design and assemble the deployment mechanism for the origami-inspired solar array shown in Figure 13. The deployment mechanism can be either external (i.e., attaches to the outside of the stowed array) or internal (i.e., built into the array itself), but it must not restrict the ability of the array to be folded into the stored configuration or interfere with the solar panels' view of the sun. All of the solar array dimensions and parameters have been defined by NASA, where the most relevant parameters are given in Table 3 and the parameter definitions are shown in Figure 14 (Copyright 2013 by ASME). For reference, these parameters are based on the full-scale solar array presented in [45], where only the first "ring" is considered. The solar array efficiency η is the percentage of incident solar radiation that is converted to usable power by the solar array.

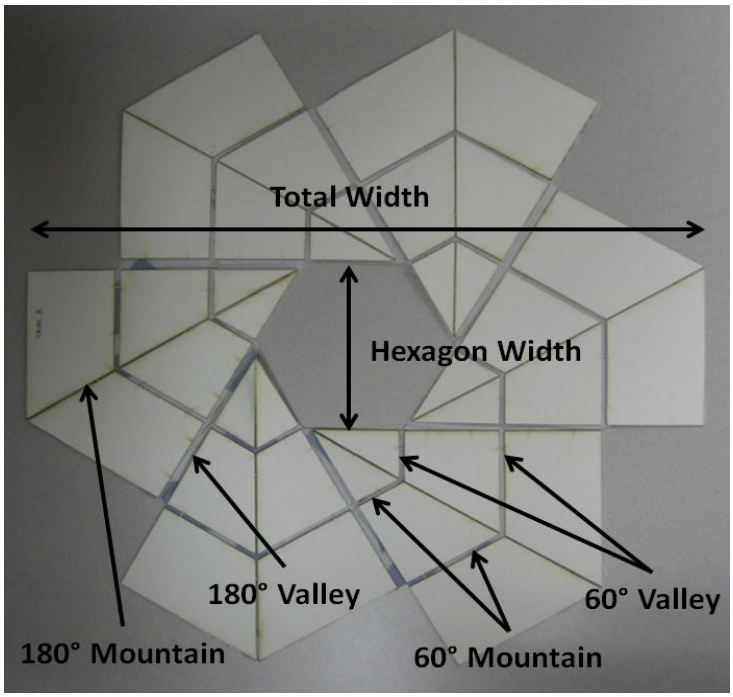

Figure 14. Solar array parameter definitions [45].

Table 3. Solar array parameters.

Parameter	Value
Total Width	10 m
Hexagon Width	2.5 m
Solar Panel Thickness	10 mm
Membrane Thickness	0.5 mm
180° Mountain Fold Spacing	0 m
180° Valley Fold Spacing	0.14 m
60° Mountain Fold Spacing	0.1 m
60° Valley Fold Spacing	0.1 m
Approximate Solar Array Surface Area	50.5 m^2
Solar Array Efficiency	50%

The white pieces in Figure 14 represent the individual solar panels for the solar array, where all of the panels are connected together by a flexible membrane. The spacing between all of the solar panels is needed in order to account for the thickness of the solar panels when in the stowed (folded) configuration. The spacing is different based on the types of folds. Each fold is either a 180° fold (adjacent panels fold completely) or a 60° fold (adjacent panels fold along the corner of the hexagon), and a mountain fold (backside of panels rotate toward each other) or valley folds (top side of panels rotate toward each other). The appropriate spacing for all of the folds is given in Table 3.

Any deployment mechanism must not interfere or prevent the array with the defined dimensions from folding both to and from the stowed configuration. In addition, any mechanism must meet the following contract requirements: a minimum deployment time from stowed configuration to open configuration of 100 seconds, a maximum mass of the entire mechanism including associated power supply of 1.5 kg, and a minimum power output of the solar array of 50 W at a solar radiation level P_d = 2 W/m^{2}s (the approximate solar radiation near the dwarf planet Pluto). If the delivered deployment mechanism meets the preceding requirements, the contract defines a payment of $1,000,000. If the delivered mechanism does not meet these requirements, no payment is guaranteed. Furthermore, the contract includes the following monetary incentive to reduce the mass as much as possible due to the high cost of launching objects into space: the payment increases at a rate of $50,000 per kg below the maximum mass (1.5 kg). Using these contract requirements, the value to the engineering firm v_f of any delivered

deployment mechanism is defined using the following equation based on the opening time t, overall mass m, and the design power output P at a solar radiation of P_d.

$$v_f = \begin{cases} \$1{,}000{,}000 + \dfrac{\$50{,}000}{kg}(1.5kg - m) & \text{if } P \geq 50W, m \leq 1.5kg, t \leq 100s \\ 0 & \text{all else} \end{cases} . (25)$$

The goal of the engineering firm is to maximize the expected value of the deployment mechanism delivered to NASA. At the current point in the design process, the firm has chosen the deployment mechanism type to be internal actuation, such that the mechanism itself is built into specific folds (or hinges) of the array. Because the solar array has only one degree of freedom, placing the actuation mechanism at one or more folds ensures that the array will unfold. The folds chosen for placement of the actuation mechanism are shown in Figure 15 marked by the 12 red circles. These folds were chosen because they require only 60° of rotation (opening from a 120° fold around the hexagon corners to 180° in the flat configuration) and provide the most versatility due to a long fold line (as compared to the parallel folds closer to the hexagon).

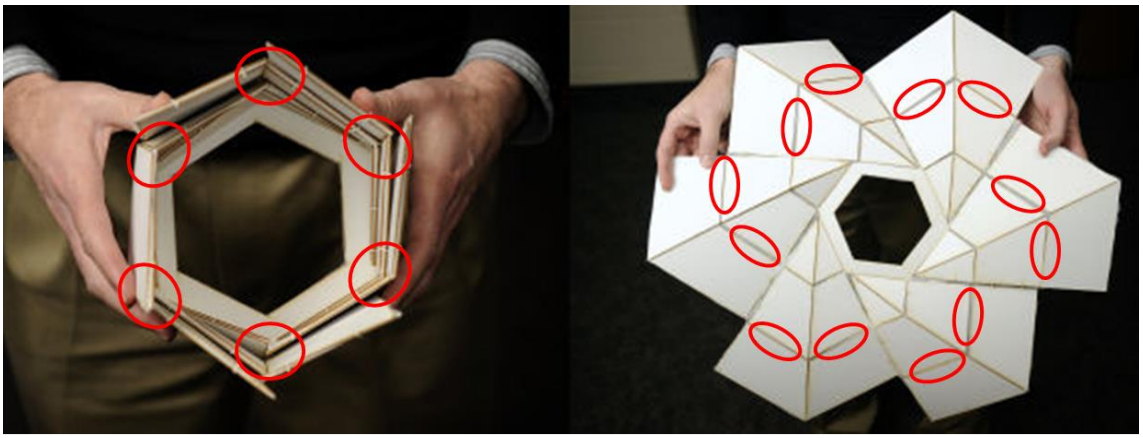

Figure 15. Actuation mechanism placement in solar array [45].

Further, the firm has also narrowed the specific mechanism deployment concept for the folds down to two: one-way actuation using torsion of Shape Memory Alloy (SMA) torque tubes and one-way actuation using bending of SMA sheets. SMA materials possess a unique characteristic that allows them to recover large amounts of strain upon sufficient heating that transforms the material from the martensitic phase to the austenitic phase. This transformation allows SMA materials to regain their original shape, called the shape-memory effect [68]. By designing the materials such that the original shape corresponds to the flat configuration, the SMA can be strained as it is folded into the stowed configuration and then later heated, causing it to transform and recover the strain to open the array.

For the SMA torque tube concept, the torque tube is placed along the length of a fold with connections at each end to opposite solar panels. Upon transformation, the SMA tube will twist, causing the panels to rotate with respect to each other and open the fold. In contrast, for the SMA sheet concept, the sheet is connected to each panel along

the length of the fold (not just at the ends as with the torque tube). Upon heating, the bent sheet will open to flat, causing the solar panels to also open flat. Both of these concepts (SMA sheet on the left, SMA torque tube on the right) are presented in Figure 16 for a single fold, where the stowed and actuated (heated) configurations are shown in parts a) and b) respectively using a section view plane with normal vector parallel to the fold line. Part c) shows the actuated configuration as viewed from above, as would be seen in Figure 15.

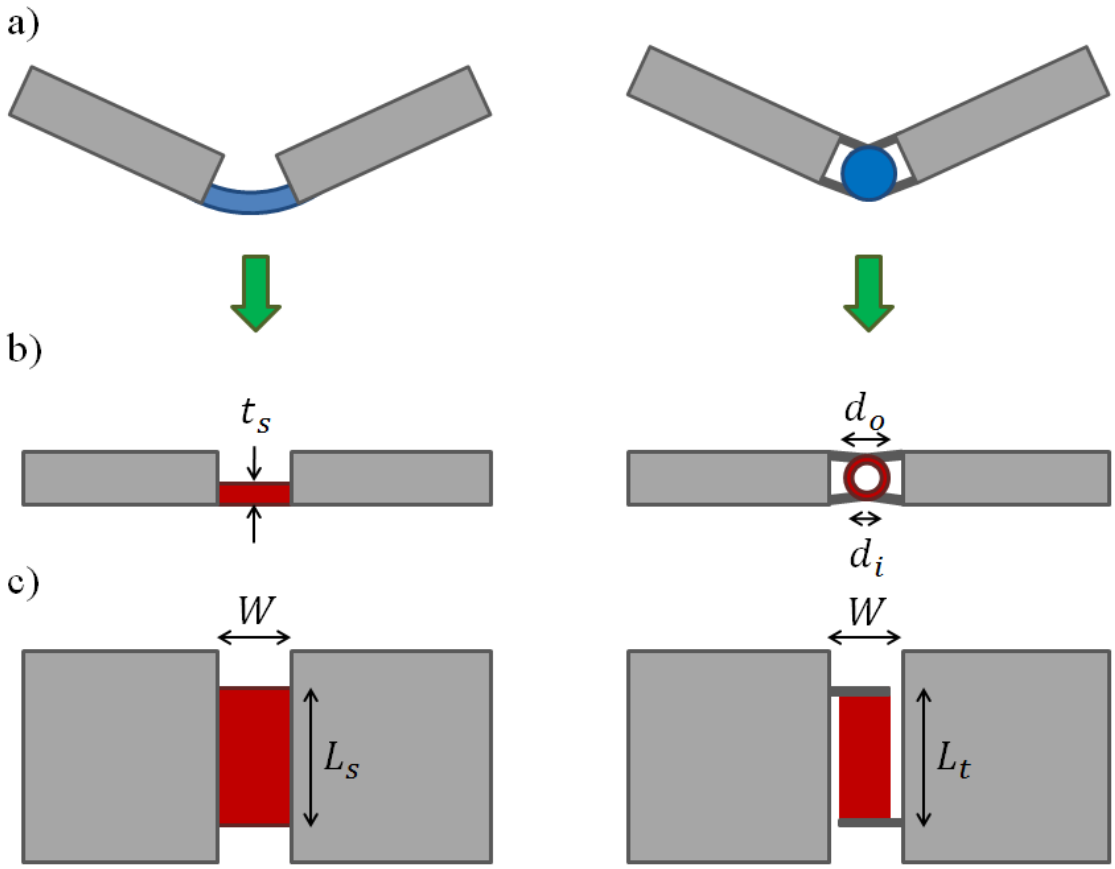

Figure 16. Stowed (a) and actuated (b,c) configurations for the SMA sheet (left) and SMA torque tube (right) design concepts.

For each of these actuation concepts, there are multiple other parameters (e.g., dimensions, connection locations, connection types, etc.) that must be chosen to create a final design. In order to avoid unnecessary complexity in this case study, the number of parameters for each design is reduced to one, treating all other parameters as constant, or already determined by the engineering firm. In addition, all 12 fold locations selected for actuation will use the exact same actuation design. As such, all subsequent design analysis will be done in terms of a single fold.

The single design parameter that the firm has left to choose for either of the design concepts is the length of the SMA segment along the fold line. This length is denoted by L_s for the SMA sheet and L_t for the SMA torque tube. The other relevant dimensional parameters are the sheet thickness t_s, the tube outer diameter d_o, tube inner diameter d_i, and the separation between the solar panels W (this is identical to the spacing between 60° mountain and valley folds of 0.1 m). These parameters are shown in Figure 16. For the SMA sheet, the sides along the length are bonded to the solar panels. For the SMA torque tube, rigid connections are applied at the two ends, where the connections are sufficiently constrained by the size of the tube and the separation between the panels.

4.2 Case Study Engineering Analysis

In order to predict the unfolding behavior of the two concepts, the firm has chosen to use pure bending and pure torsion analysis assuming a constant applied

moment that opposes the unfolding due to the stiffness of the non-actuated connections between panels in the solar array (while in general more detailed and rigorous analysis methods would likely be used in practice, such as finite element analysis, the basic analysis presented here is sufficient for this case study).

Although the engineering analysis for this case study is basic, the derivation of the equations for the unfolding behavior of the two design concepts before and after actuation is lengthy because of the presence of SMA material. This is due to the complex constitutive behavior of SMAs during phase transformation. As such, this derivation is presented in the Appendix. The result of this derivation is the unfolding behavior of each concept in terms of the fold angle θ between adjacent panels in the solar array as shown in Figure 17, where an angle of 180° corresponds to the solar array opening completely flat. An angle less than 180° reveals that the solar array has not opened completely.

Using this analysis, the firm can predict the extent to which the solar array opens to the flat configuration, which directly determines the power output capability of the solar array. This power output is calculated by determining the reduction in the amount of surface area that is facing the sun A_s. In other words, this is determined by the reduction in the footprint of the solar array for harvesting solar energy. Using the total solar array surface area A_0 and solar array efficiency η given in Section 4.1, the total power provided by the solar array at P_d can be calculated as a function of A_s (see Figure 17):

$$A_s = \begin{cases} A_0, & \theta = 180 \\ A_0 \cos(90° - \frac{\theta}{2}), & \theta < 180 \end{cases}, \tag{26}$$

$$P = P_d A_s \eta. \tag{27}$$

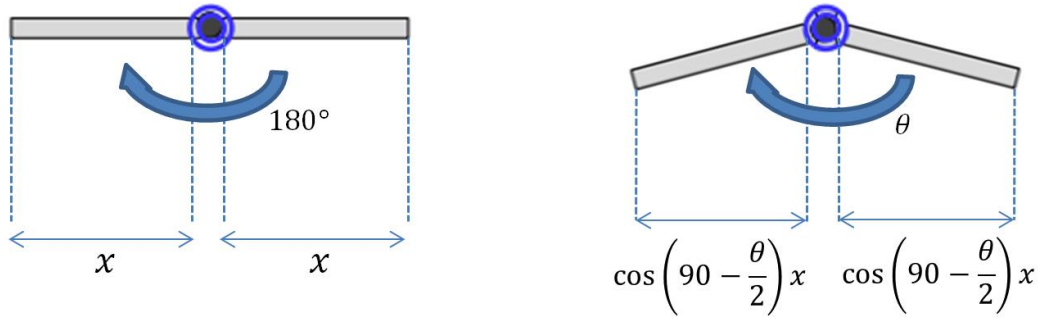

Figure 17. Demonstration of reduction in surface area for partially opened fold.

With the power analysis of any solar array design completed, the mass of the design must be calculated. The firm has recognized that the overwhelming factor in determining the mass of the entire actuation mechanism for the two proposed SMA concepts is the battery size required to provide the necessary power to heat the SMA within the time frame required (100 s). The SMA sheet and SMA torque tubes appropriate for this solar array are of such small size that their total mass is ~2 orders of magnitude lower than the mass required in batteries. The firm has also recognized that the power required to heat the SMA is minimized by using the maximum allowed deployment time. As such, the analysis for calculating the power required for the actuation concepts assumes an actuation time of 100 s.

Again, because of the complex transformation behavior of SMA materials, the derivation of the power required to actuate the SMA design concepts is lengthy. Thus, the majority of this derivation is also presented in the Appendix, where the result is the

energy required to fully actuate the SMA concepts, denoted by E. By assuming a constant applied power by the batteries over the 100 s time frame, the required battery power can be calculated. NASA has required that any batteries used be a specific Lithium Ion battery that has a mass power density P_m. By insulating the SMA material (i.e., assuming adiabatic conditions), the mass m required in Lithium Ion batteries to heat the SMA is calculated as follows:

$$m = \frac{E}{P_m t_o}. \tag{28}$$

4.3 Information-gathering Decision Problem Definition

The previous section defines the firm's calculation of power output and mass for any design, if all material properties and relevant parameters are known with *certainty*. This means that the value of any design is solely a function of the power output and mass, which in turn are directly related to the length of the design and the choice of SMA sheet or SMA torque tube: $v_f(L_s)$ or $v_f(L_t)$. For both designs, the preceding analysis demonstrates that a minimum length (L_s or L_t) is desired to reduce the mass of the mechanism. However, a design that is too short results in too much strain such that SMA transformation cannot recover enough strain to make the array open to flat. The objective of the firm is to choose a design that maximizes value V_f:

$$V_f = \max_{L_t, L_s} v_f = \max\left(\max_{L_t} v_f(L_t,)\,, \max_{L_s} v_f(L_s)\right). \tag{29}$$

However, the engineering firm does not know all of the SMA material properties with certainty. Specifically, the firm has uncertainty in the maximum transformation strain values H_t and H_s. These properties determine the maximum amount of recoverable strain the SMA material can achieve, which have a direct effect on whether or not the actuation mechanisms can unfold fully. Because of this uncertainty, the firm cannot know the exact power output of any design. As such, the value of any design is uncertain, requiring the firm's value function to be a function of both L and H, and the firm's objective to be: choose a design that maximizes the *expected* value $E[V_f]$:

$$\mathrm{E}[\mathrm{V_f}] = \max\left(\max_{L_t} \underset{H_t}{E}\left[v_f(H_t, L_t)\right], \max_{L_s} \underset{H_s}{\mathrm{E}}\left[v_f(H_s, L_s)\right]\right). \tag{30}$$

In order to reduce this uncertainty, the firm can conduct thermo-mechanical testing to gather information about either H_s or H_t for the SMA, yielding two separate information-gathering actions t_s and t_t respectively. The individual costs of the tests for the SMA sheet and SMA torque tube are denoted by C_s and C_t. The accuracy of the testing method is characterized by a normal distribution with a standard deviation in the value of H denoted by σ_s and σ_t (this is presented mathematically later in this section). The engineering firm must decide whether to choose a design concept and SMA component length, or pursue the information-gathering action. If a testing option is chosen, an observation is received and the firm returns to the same decision with an updated belief in the appropriate uncertain parameter and incurs the associated cost. The firm should only choose one of the information-gathering actions if it is expected to increase the value of the final chosen design. While it is possible for the firm to gather

information on both parameters, the sequential analysis dictates that only one action be chosen first. The other information-gathering action can be chosen at the next step if it is still expected to be valuable to do so after the information received from the first action.

In order to represent this problem in the POMDP formalism, the continuous uncertain parameters H_s and H_t and design variables L_s and L_t must be discretized to a finite number of intervals. To do this, a representative range for each must be chosen and discretized into a varying number of intervals. The numeric value of each interval represents the midpoint of the interval and w defines the size of each interval, where the number of intervals N is directly related to w by $N = \frac{max-min}{w} + 1$. This results in the following definition of the discretization:

$$H_s = \left\{\left[H_s^{min} : w_{H_s} : H_s^{max}\right]\right\} = \left\{H_s^1, H_s^2, \dots H_s^{N_{H_s}}\right\}, \tag{31}$$

$$H_t = \left\{\left[H_t^{min} : w_{H_t} : H_t^{max}\right]\right\} = \left\{H_t^1, H_t^2, \dots H_t^{N_{H_t}}\right\}, \tag{32}$$

$$L_s = \left\{\left[L_s^{min} : w_{L_s} : L_s^{max}\right]\right\} = \left\{L_s^1, L_s^2, \dots L_s^{N_{L_s}}\right\}, \tag{33}$$

$$L_t = \left\{\left[L_t^{min} : w_{L_t} : L_t^{max}\right]\right\} = \left\{L_t^1, L_t^2, \dots L_t^{N_{L_t}}\right\}. \tag{34}$$

As noted in Section 3, in this work there is one observation for each possible value of the uncertain parameters. Following the variable naming and definition convention presented in Section 3, the relevant POMDP parameters are defined as follows, where the state space for this problem must include the value of both H_s and H_t in every state:

$$S = \left\{(H_s^1, H_t^1), (H_s^2, H_t^1), \dots \left(H_s^{N_{H_s}}, H_t^1\right), (H_s^2, H_t^1), (H_s^2, H_t^2), \dots \left(H_s^{N_{H_s}}, H_t^{N_{H_t}}\right), Z\right\}$$

$$= \left\{s_1, s_2, \dots s_{N_{H_s} * N_{H_t} + 1}\right\}, \tag{35}$$

$$A = \left\{L_s^1, L_s^2, \dots L_s^{N_{L_s}}, L_t^1, L_t^2, \dots L_t^{N_{L_t}}, t_s, t_t\right\} = \left\{a_1, a_2, \dots a_{N_{L_s} + N_{L_t} + 2}\right\}, \tag{36}$$

$$\Omega = \{H_s^1, H_s^2, \dots H_s^{N_{H_s}}, H_t^1, H_t^2, \dots H_t^{N_{H_t}}\} = \{o_1, o_2, \dots o_{N_{H_s} + N_{H_t}}\}. \tag{37}$$

The transition function is defined as in Section 3. The observation function must represent the normal distribution for the information-gathering actions. Because of the discrete intervals used to represent H, the normal distribution must be discretized. Further, each information-gathering action can only result in the corresponding observations for that action; action t_s can only give observations $\{o_1, \dots o_{N_{H_s}}\}$ while action t_t can only give the remaining observations (this simply represents the fact that running the thermo-mechanical test for H_s does not reveal an observation about H_t). As such, the observation function $O(a, s', o)$ for the information-gathering actions are defined as follows for $i, j = 1{:}\, N_{H_s}$ and $k, l = 1{:}\, N_{H_t}$:

$$O\left(t_s, \left(H_s^i, H_t^k\right), H_s^j\right) = C_1 \int_{H_s^j - 0.5 w_{H_s}}^{H_s^j + 0.5 w_{H_s}} \frac{1}{\sigma_s \sqrt{2\pi}} e^{-\frac{\left(x - H_s^i\right)^2}{2\sigma_s^2}} dx, \tag{38}$$

$$O\left(t_t, \left(H_s^i, H_t^k\right), H_t^l\right) = C_2 \int_{H_t^l - 0.5 w_{H_t}}^{H_t^l + 0.5 w_{H_t}} \frac{1}{\sigma_t \sqrt{2\pi}} e^{-\frac{\left(x - H_t^k\right)^2}{2\sigma_t^2}} dx, \tag{39}$$

$$O\left(t_s, Z, H_s^j\right) = O\left(t_t, Z, H_t^l\right) = \frac{1}{N}, \tag{40}$$

$$C_1 = \frac{1}{\int_{H_s^1 - 0.5w_{H_s}}^{H_s^{N_{H_s}} + 0.5w_{H_s}} \frac{1}{\sigma_s\sqrt{2\pi}} e^{-\frac{\left(x - H_s^i\right)^2}{2\sigma_s^2}} dx}, \tag{41}$$

$$C_2 = \frac{1}{\int_{H_t^1 - 0.5w_{H_t}}^{H_t^{N_{H_t}} + 0.5w_{H_t}} \frac{1}{\sigma_t\sqrt{2\pi}} e^{-\frac{\left(x - H_t^k\right)^2}{2\sigma_t^2}} dx}, \tag{42}$$

where all other values of $O(a, s', o)$ are equal to 0 and C_1 and C_2 are normalizing factors to ensure that the probability of all observations sum to 1. For the rewards, if the current state is the absorbing state, the reward is set to 0 for all actions: $R(Z, a) = 0$. The rewards for the information-gathering actions are defined by the cost of the tests and are not a function of the current state. The rewards for the testing actions and commitment actions (choosing a design) are defined as follows for $m = 1{:}\, N_{L_s}$ and $n = 1{:}\, N_{L_t}$:

$$R\left(\left(H_s^i, H_t^k\right), t_s\right) = -C_s, \tag{43}$$

$$R\left(\left(H_s^i, H_t^k\right), t_t\right) = -C_t, \tag{44}$$

$$R\left(\left(H_s^i, H_t^k\right), L_s^m\right) = v_f\left(H_s^i, L_s^m\right), \tag{45}$$

$$R\left(\left(H_t^i, H_t^k\right), L_t^n\right) = v_f\left(H_t^k, L_t^n\right). \tag{46}$$

According to the previous problem definition and POMDP parameter definition, all constant parameters for this case study are presented in Table 4, where the SMA material properties are based on values from [69, 70].

Table 4. Case study parameters.

Parameter	Value
Sheet Thickness, t	3 mm
Tube Outer Diameter, d_o	10mm
Tube Inner Diameter, d_i	9 mm
Sheet Design Length Range, L_s	[6 mm, 10 mm]
Tube Design Length Range, L_t	[12 cm, 20 cm]
Maximum Sheet Transformation Strain Range, H_s	[3.7%, 4.7%]
Maximum Tube Transformation Strain Range, H_t	[2.2%, 3.2%]
Applied Moment, M	10 N-m
SMA Elastic Modulus, E	35 GPa
SMA Shear Modulus, G	13 GPa
SMA Stress Influence Coefficient of Austenite, C^A	9 MPa/K
SMA Austenitic Start Temperature, A_s	342 K
SMA Austenitic Finish Temperature, A_f	367 K
SMA Martensitic Start Temperature, M_s	327 K
SMA Martensitic Finish Temperature, M_f	302 K
Lithium Ion Mass Power Density, P_m	250 W/kg
SMA Density, ρ_{SMA}	6450 kg/m^3
SMA Specific Heat Capacity, c_p	329 J/kgK
Initial Temperature of SMA in Space, T_0	5 K
Fully Transformed Temperature of SMA, T_f	400 K
Cost of Both Thermo-Mechanical Tests, C_s and C_t	$1,500
Standard Deviation of Both Tests, σ_s and σ_t	0.1%

In order to gain a better perspective of the parameters for this case study, it is helpful to present the solution to the decision problem when the values of the uncertain parameters are known with certainty. Consider the case where $N_{H_s} = N_{H_t} = N_{L_s} = N_{L_t} = 1$, which gives 10 possible values of each uncertain parameter and 10 design

lengths for each concept. Because there are 100 combinations for the possible values of H_s and H_t, the results are best shown using two separate tables, one for each concept. These are presented on the following page where each table presents the value of choosing any of the design alternatives (L_s or L_t) for any values of the uncertain parameter H_s and H_t.

In order to determine the best design alternative for any combination of H_t and H_s known with certainty, locate the appropriate columns in each table (Table 5 for H_t and Table 6 for H_s). Within each column, find the row with the highest value and the corresponding length for that row. Compare the largest values from the two tables and take the design concept with the higher value. This is the concept, and associated length, that maximizes the value of the design. For example, consider $H_t = 4.58$ and $H_s = 4.48$. The appropriate cells for the maximum value in each table are boxed. According to these values, the best design is the SMA sheet with a length of 6.67 mm, which has a value of 1.027 million dollars.

Table 5. Certain design values (in millions) of SMA torque tube concept for any combination of L_t and H_t.

L_t, mm \ H_t, %	3.81	4.00	4.20	4.39	4.58	4.77	4.97	5.16	5.35	5.54
120	0	0	0	0	0	0	0	0	0	1.035
129	0	0	0	0	0	0	0	0	1.032	1.032
138	0	0	0	0	0	0	1.029	1.029	1.029	1.029
147	0	0	0	0	0	1.027	1.026	1.026	1.026	1.026
156	0	0	0	0	1.024	1.024	1.023	1.023	1.023	1.023
164	0	0	0	1.021	1.021	1.021	1.021	1.020	1.020	1.020
173	0	0	1.018	1.018	1.018	1.018	1.018	1.017	1.017	1.017
182	0	0	1.016	1.015	1.015	1.015	1.015	1.014	1.014	1.014
191	0	1.013	1.013	1.012	1.012	1.012	1.012	1.011	1.011	1.011
200	1.010	1.010	1.010	1.009	1.009	1.009	1.009	1.009	1.008	1.008

Table 6. Certain design values (in millions) of SMA sheet concept for any combination of L_s and H_s.

L_s, mm \ H_s, %	3.7	3.81	3.92	4.03	4.14	4.26	4.37	4.48	4.59	4.7
6	0	0	0	0	0	0	0	0	0	0
6.33	0	0	0	0	0	0	0	0	1.029	1.029
6.67	0	0	0	0	0	0	0	1.027	1.027	1.027
7	0	0	0	0	0	0	1.025	1.025	1.025	1.024
7.33	0	0	0	0	0	1.023	1.023	1.023	1.022	1.022
7.67	0	0	0	0	1.021	1.021	1.020	1.020	1.020	1.020
8	0	0	0	1.019	1.019	1.018	1.018	1.018	1.018	1.017
8.33	0	0	1.017	1.016	1.016	1.016	1.016	1.016	1.015	1.015
8.67	0	1.015	1.014	1.014	1.014	1.014	1.013	1.013	1.013	1.013
9	1.013	1.012	1.012	1.012	1.012	1.011	1.011	1.011	1.011	1.010

4.4 Case Study Comparison Methods

As mentioned in Section 1, most available methods used for information-gathering decision making make many simplifying assumptions and do not explicitly model both the sequential and imperfect nature of information-gathering decisions due to the inherent complexity required to do so. The POMDP formalism is capable of achieving this explicit modeling, but it has yet to be shown that this modeling is of significant value to a DM. In order to achieve this, the POMDP formalism is compared to Expected Value of Information (EVI) theory for the information-gathering decision problem in this case study. Specifically, the Expected Value of Perfect Information (EVPI), the sequential Expected Value of Partial Perfect Information (sEVPPI), and the Expected Value of Sample Information (EVSI) are used for comparison with the POMDP [17, 71]. While in practice most EVI methods are used informally or heuristically as described in Section 1, they still provide a baseline for comparison to demonstrate the benefit of using an explicit modeling framework such as the POMDP formalism. In this portion of the case study, all POMDP solutions are generated using Perseus.

The definitions of the EVI methods for comparison are presented here in the context of the specific case study design problem. For all methods, the expected value of information is defined as the difference between the expected value of choosing a design after gathering information (thus a design is chosen with an updated belief) and the expected value of choosing a design alternative at the initial belief of the DM. The

associated costs of gathering the information are not considered in the calculation of the expected value, but in the resulting decision rule that follows (described later in this section). The expected value of choosing a design at the current belief $E[V_f]$ is calculated as follows:

$$E\left[V_f\right] = \max_{L_t, L_s} \underset{H_t, H_s}{E} [\, v^*(H_t, H_s, L)]$$

$$= \max\left(\max_{L_t} \underset{H_t}{E}\left[v_f(H_t, L_t)\right], \max_{L_s} \underset{H_s}{E}\left[v_f(H_s, L_s)\right]\right). \tag{47}$$

Note that by the previous equation, a new version of the firm's value function has been defined: $v_f(H, L) = v^*(H_t, H_s, L)$, where the only change is that both values of H are included as inputs. For example, if the length input is L_t, the input H_s only acts as a placeholder and does not affect the output value. This is done as a matter of convenience to make the representation of the following EVI methods more compact.

EVPI provides the expected value of gathering information about all available uncertain parameters if the information-gathering action(s) provide perfect information about the associated uncertain parameters:

$$EVPI = \underset{H_t, H_s}{E} [\max_{L_t, L_s} v^*(H_t, H_s, L)] - E\left[V_f\right]. \tag{48}$$

Because the information is perfect, information is only gathered once and then the DM chooses a design alternative. This assumes that the DM will no longer have uncertainty in any parameters and can accordingly make the best choice of design alternative. Similarly, the Expected Value of Partial Perfect Information (EVPPI) also

provides the expected value of perfect information but it allows perfect information to be gathered about only one of the uncertain parameters. Again, information can only be gathered once. As such, EVPPI is defined for each of the information-gathering actions in this case study as follows:

$$EVPPI = \begin{cases} E_{H_s}\left(\max\limits_{L_s,L_t} \underset{H_t}{E}[\,v^*(H_t,H_s,L)]\right) - E[V_f] & \text{for } t_s \\ E_{H_t}\left(\max\limits_{L_s,L_t} \underset{H_s}{E}[\,v^*(H_t,H_s,L)]\right) - E[V_f] & \text{for } t_t \end{cases}. \quad (49)$$

It is also possible to apply EVPPI sequentially by choosing to gather perfect information about one uncertain parameter, and then, depending on the information received, choosing whether or not to gather perfect information about the other uncertain parameter(s) [71]. The mathematical definition of this method is presented later in this section.

In contrast to these methods that assume perfect sources of information, EVSI incorporates imperfect sources of information. Because the information is imperfect, it is feasible to gather information more than once and collect multiple "samples" or observations. As described in Section 1, EVSI calculates the number of "samples" that should be taken in addition to calculating the associated expected value. Even though EVSI allows for information to be gathered multiple times, it is not a sequential analysis method: the DM must choose the number of samples to gather at the onset of the decision. While in general, EVSI is defined in terms of a single uncertain parameter with one source of information, it can be expanded to two uncertain parameters and information-gathering actions:

$$EVSI = \max_{n_s,n_t}\left(\underset{z}{E}\left[\max_{L_s,L_t}\left(\underset{(H_t,H_s|z)}{E}[\,v^*(H_t,H_s,L)]\right)\right] - E\left[V_f\right]\right), \tag{50}$$

$$z = \left\{z_s^1, z_s^2, \ldots z_s^{n_s}, z_t^1, z_t^2, \ldots z_t^{n_t}\right\}, \tag{51}$$

where n_s and n_t are the number of times each information-gathering action is executed over the range $[0, \infty)$, z is the set of observations received from the information-gathering actions about H_s and H_t, and $\underset{(H_T,H_B|z)}{E}$ is the expectation over the uncertain parameters given the updated belief based on the observation vector z. As a summary of the defined EVI approaches, Table 7 presents the high level capabilities of each approach, compared to the POMDP approach, in terms of the types of information sources that can be represented. Here, "Partial Information Sources" refers to the ability of the method to incorporate actions that only provide information about a portion of the uncertain parameter space.

Table 7. Comparison of EVI approaches and POMDP.

EVI Method	Imperfect Information Sources?	Partial Information Sources?	Repeatable Information Sources?	Sequential Information Sources?
EVPI	No	No	No	No
EVPPI	No	Yes	No	No
sEVPPI	No	Yes	No	Yes
EVSI	Yes	Yes	Yes	No
POMDP	Yes	Yes	Yes	Yes

The definitions of the EVI methods presented here only describe approximate methods for calculating the expected value of gathering information. They do not generate a *policy* for the DM to follow, which is needed for comparison to the POMDP. The POMDP solution, a policy, provides the DM with the most valuable action to take as well as the expected value of taking that action. In order to compare the POMDP formalism to the EVI methods, a policy for each method must be created. Again, most EVI methods, are used heuristically. For example, because EVPI calculates the value of gathering perfect information, it is often used as an upper bound on what the DM should pay for gathering information; if it is not valuable to gather information even if the information were perfect, it must then not be valuable to gather imperfect information.

For the purposes of this case study, a decision policy is generated for each of the EVI methods as follows, where the policy defines the current action to take as well as the action to be taken following information-gathering:

$$\text{EVPI Policy: } \begin{cases} \textbf{Perform } \boldsymbol{t_s} \textbf{ and } \boldsymbol{t_t}, & \textbf{for } \boldsymbol{EVPI \cdot w - C_s - C_t > 0} \\ \quad \rightarrow \text{Followed by } L_d & \\ \textbf{Choose } \boldsymbol{L_d}, & \textbf{for } \boldsymbol{EVPI \cdot w - C_s - C_t \leq 0} \end{cases}, \quad (52)$$

$$\text{EVSI Policy: } \begin{cases} \textbf{Perform } \boldsymbol{t_s}, * \boldsymbol{n_s} \textbf{ and } \boldsymbol{t_t} * \boldsymbol{n_t}, & \textbf{for } \boldsymbol{EVSI - n_s C_s - n_t C_t > 0} \\ \quad \rightarrow \text{Followed by } L_d & \\ \textbf{Choose } \boldsymbol{L_d}, & \textbf{for } \boldsymbol{EVSI - n_s C_s - n_t C_t \leq 0} \end{cases}, \quad (53)$$

where the action L_d is the selection of the design alternative that maximizes the expected value based on the current belief, and $w \epsilon (0,1]$ is a weighting factor. The weighting factor is included to account for the fact that EVPI represents an upper bound. In

general, a DM would not choose to gather information solely based on the value of perfect information, because they only have access to imperfect information. As such, w serves to scale down EVPI to be more representative of the true problem. It is important to note that L_d is chosen using the *current belief*, which is a function of the observations received from gathering information (if any has been gathered) via Bayes rule (see Section 3.1).

EVPPI is not included because it only computes the expected value of gathering information about each uncertain parameter individually. This can be addressed by using sEVPPI instead. Unlike the other EVI methods, sEVPPI must include the cost of the tests in the formulation, as well as the decision policy. This is because the corresponding expected value is dependent upon the choice of gathering information in the future. Using the weighting factor from EVPI, sEVPPI and the appropriate decision policy are defined for this case study as follows, where "$Followed\ by$" is abbreviated by "$F.B.$":

$$sEVPPI = \begin{cases} E_{H_s}\left[\max\left\{\underset{H_t}{E}[\max_{L_s,L_t} v^*(H_t,H_s,L)]w - C_t\,,\ \max_{L_s,L_t}\underset{H_t}{E}[\,v^*(H_t,H_s,L)]\right\}\right] w - C_s - E[V_f] & \text{for } t_s \\ E_{H_t}\left[\max\left\{\underset{H_s}{E}[\max_{L_s,L_t} v^*(H_t,H_s,L)]w - C_s\,,\ \max_{L_s,L_t}\underset{H_s}{E}[\,v^*(H_t,H_s,L)]\right\}\right] w - C_t - E[V_f] & \text{for } t_t \end{cases}, \quad (54)$$

$$sEVPPI \text{ Policy: } \begin{cases} \textbf{Perform } \boldsymbol{t_s}, \quad \textbf{for } \boldsymbol{sEVPPI(t_s) > sEVPPI(t_t) > 0} \\ \quad \rightarrow \text{F.B. } t_t, \quad \text{for } \underset{H_t}{E}[\max_{L_s,L_t} v^*(H_t,H_s,L)]w - C_t > \max_{L_s,L_t}\underset{H_t}{E}[\,v^*(H_t,H_s,L)] \\ \qquad\quad \rightarrow \text{F.B. } L_d \\ \quad \rightarrow \text{F.B. } L_d, \quad \text{for } \underset{H_t}{E}[\max_{L_s,L_t} v^*(H_t,H_s,L)]w - C_t \le \max_{L_s,L_t}\underset{H_t}{E}[\,v^*(H_t,H_s,L)] \\ \textbf{Perform } \boldsymbol{t_t}, \quad \textbf{for } \boldsymbol{sEVPPI(t_t) > sEVPPI(t_s) > 0} \\ \quad \rightarrow \text{F.B. } t_s, \quad \text{for } \underset{H_s}{E}[\max_{L_s,L_t} v^*(H_t,H_s,L)]w - C_s > \max_{L_s,L_t}\underset{H_s}{E}[\,v^*(H_t,H_s,L)] \\ \qquad\quad \rightarrow \text{F.B. } L_d \\ \quad \rightarrow \text{F.B. } L_d, \quad \text{for } \underset{H_s}{E}[\max_{L_s,L_t} v^*(H_t,H_s,L)]w - C_s \le \max_{L_s,L_t}\underset{H_s}{E}[\,v^*(H_t,H_s,L)] \\ \textbf{Choose } \boldsymbol{L_d}, \quad \textbf{for } \boldsymbol{0 \ge sEVPPI(t_t) > sEVPPI(t_s)} \end{cases}, \quad (55)$$

Using the policy definitions presented, an exact policy can be computed as a function of the appropriate EVI method, which is a function of the DM's initial belief b_0; each policy is unique to the specific initial belief. The policy prescribes the initial action to take, as well as the action to take at subsequent steps as a function of the observation(s) received. Because of this, the policy directs the DM all the way through to the selection of a design alternative.

With the decision policy defined for all three EVI methods, a direct metric of comparison must be established to compare the execution of these policies to the POMDP solution policy. In order to do so, each policy can be applied to the full, explicit decision tree of the information-gathering decision problem. By doing so, the associated expected values of executing each policy can be compared. As mentioned in Section 1, solving such a tree can be computationally prohibitive in practice. However, for this academic exercise, the high computational cost is acceptable purely for establishing a metric of comparison for the available policies. This decision tree contains all possible combinations of actions (both commitment actions and information-gathering actions), observations, and true states of the uncertain variables. This provides a direct comparison between methods because each policy is applied to the exact same decision tree, where the differences in the policies cause the DM to traverse the tree differently. For each policy, the expected value of executing that policy on the decision tree can be computed, and then compared to the other policies to determine which policy brings the highest expected value to the DM.

For example, using the sample decision tree presented in Section 2.2, let the decision polices for EVPI and POMDP be the following for some belief b_0:

EVPI: **Choose Design A**

POMDP: **Gather Information**
→ if Obs 1 Received, Choose Design A
→ if Obs 2 received, Choose Design B

These decision policies are presented visually in Figure 18 using bolded lines to indicate the decisions for each policy.

Using the abbreviated variables defined in Section 2.2, the expected value of following the EVPI policy and the POMDP policy, E_{EVPI} and E_{POMDP} respectively, are computed as follows:

$$E_{EVPI} = \Pr(1)\mathrm{R}(A,1) + \Pr(2)\mathrm{R}(A,2), \tag{56}$$

$$E_{POMDP} = \Pr(1)[\Pr(1|1)\mathrm{R}(A,1) + \Pr(2|1)\mathrm{R}(A,2)] + \Pr(2)[\Pr(1|2)\mathrm{R}(B,1) + \Pr(2|2)\mathrm{R}(B,2)] - \mathrm{Cost}. \tag{57}$$

Using this method, the expected value of executing each of the EVI policies and the POMDP policy can be evaluated and compared. This comparison process is presented graphically in Figure 19.

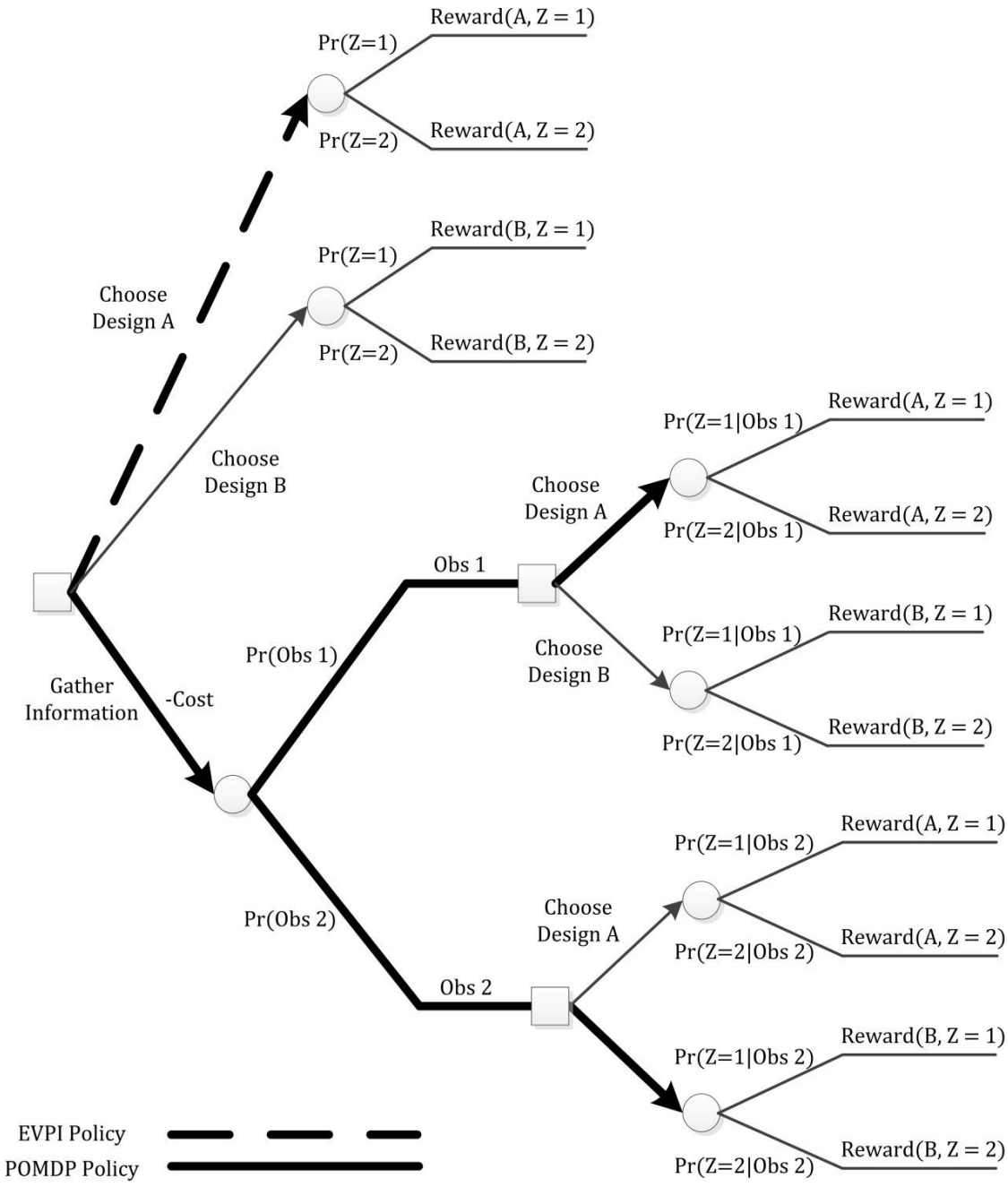

Figure 18. Visual representation of EVPI and POMDP policies for example decision tree.

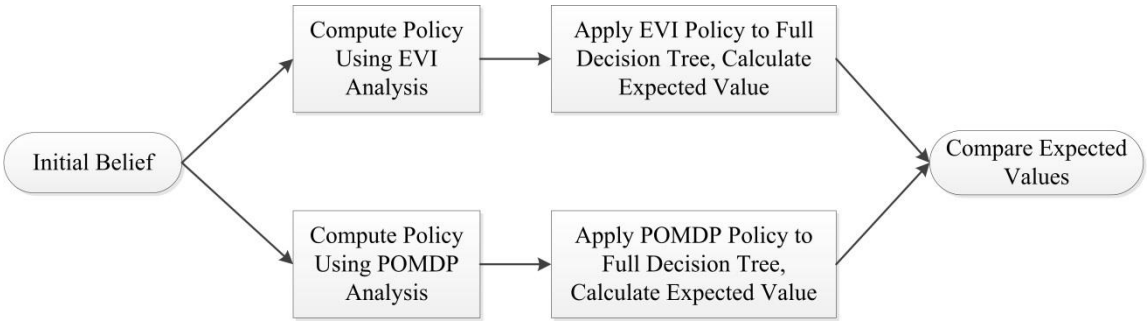

Figure 19. Comparison procedure for all EVI methods and POMDP.

Without regard to the cost of generating the policies, a higher expected value represents the decision analysis method that is more preferred. However, each of these policies has a different computational complexity, or resource cost. It is important to note that both the EVI methods and the POMDP framework require the same input information about the information-gathering decision problem. Because of this, the computational complexity of generating the associated policy is the main factor that differentiates the resource costs of implementing the different methods. A decision method that increases the expected value of the decision process is only beneficial if the additional computational cost does not negate the increase in expected value.

As a means of comparing the computational complexity of the POMDP to the EVI methods, the computation time required to generate the policy for each method is recorded. The EVPI, EVSI, and sEVPPI policies are computed exactly, without the use of an approximate, and possibly more efficient, methods. While it is possible that more sophisticated, approximate solution methods are available for these EVI methods,

computing the exact solution is sufficient for an initial comparison to the POMDP approach using the Perseus algorithm.

For this work, computation time is measured in terms of the wall-clock time required to compute the solution for each algorithm, similar to the POMDP solution method comparison in [54]. All computation is performed using MATLAB R2013a on a single machine for a fair comparison: a 3.4Ghz Intel(R) Core i7-2600 CPU with 8GB of system memory. However, in order to reduce the potential random effects of the system (e.g., background processes occupying computational resources) on the wall-clock time as well as the inherent stochastic nature of Perseus, each solution is calculated 10 times and averaged.

4.5 Comparison Results

For the comparison of the POMDP approach (using Perseus) to the EVI methods, first a single problem size was chosen to compare the expected values of several different initial beliefs of the DM. The problem size chosen is 101 states (each uncertain parameter discretized to 10 possible values) and 22 actions (each of the two design concepts discretized to 10 design alternatives): $N_{H_s} = N_{H_t} = N_{L_s} = N_{L_t} = 10$. In addition to the uniform belief, the initial beliefs chosen for this comparison include cases where the DM's uncertainty in one of the two uncertain parameters is larger than the other. Recall from Section 3.3 that a partial belief is a belief over a single uncertain parameter. Using this definition, Table 8 designates three partial beliefs for use in this analysis, where each is either uniform over the entire range of the uncertain parameter or

uniform over half of the range. The initial beliefs of the DM chosen for this case study are combinations of these partial beliefs, where each of the two uncertain parameters takes on one of the available partial beliefs in Table 8. Recall that all the elements of any belief must sum to 1 (see Section 2.1.2). The combinations chosen are given in Table 9. The uncertain parameter values are included from Section 6.1 for clarity for the problem size chosen.

Table 8. Partial belief definitions

Partial Belief Designation	Partial Belief
Partial Belief 1, PB1	0.1*[1,1,1,1,1,1,1,1,1,1]
Partial Belief 2, PB2	0.2*[1,1,1,1,1,0,0,0,0,0]
Partial Belief 3, PB3	0.2*[0,0,0,0,0,1,1,1,1,1]

Table 9. Belief definitions based on partial beliefs

Belief Designation	Partial Belief over H_s ($H_s = [3.7\%: 0.1\%: 4.7\%]$)	Partial Belief over H_t ($H_t = [2.2\%: 0.1\%: 3.2\%]$)
Belief 1, B1	PB1	PB1
Belief 2, B2	PB2	PB1
Belief 3, B3	PB3	PB1
Belief 4, B4	PB1	PB2
Belief 5, B5	PB1	PB3

For each of the initial beliefs defined above, the EVI methods and Perseus are used to solve for the optimal policy. For Perseus, the algorithm parameters are defined as

follows: the number of sample beliefs used to approximate the belief space is 100,000, the discount factor γ is $1 - 1x10^{-3}$ and the convergence criterion ϵ is $1x10^{-6}$. The expected value of following each policy is calculated as described in Section 4.4. For the EVPI and sEVPPI methods, three different values of the weighting factor w are used (1, 0.75, and 0.5), yielding three different policies each, for total of 9 different polices to be applied to each of the 5 initial beliefs. The results for all policies and initial beliefs are shown in Table 10, where each expected value shown is the difference of the respective method relative to the expected value of choosing a design alternative without gathering information. The results are grouped in order to compare the different EVI methods as well as compare the same EVI method with different weights.

Table 10. Increase in expected value when following EVI methods and POMDP.

Method	Expected Value Relative to No Information				
	B1	**B2**	**B3**	**B4**	**B5**
EVPI, ($w = 1$)	\$2,924.67	\$1,762.73	(\$953.27)	(\$5.33)	0*
EVSI	\$3,205.19	\$3,205.19	\$85.88	\$1,136.84	0*
sEVPPI, ($w = 1$)	\$3,501.57	\$3,048.94	(\$713.82)	\$717.05	0*
Perseus	\$4,108.15	\$3,678.53	\$266.59	\$1,372.48	\$0.00
EVPI, ($w = 1$)	\$2,924.67	\$1,762.73	(\$953.27)	(\$5.33)	0*
EVPI, ($w = 0.75$)	\$2,924.67	\$1,762.73	(\$953.27)	(\$5.33)	0*
EVPI, ($w = 0.5$)	\$2,924.67	\$1,762.73	0*	(\$5.33)	0*
sEVPPI, ($w = 1$)	\$3,501.57	\$3,048.94	(\$713.82)	\$717.05	0*
sEVPPI, ($w = 0.75$)	\$3,501.57	\$3,501.57	(\$2,435.02)	\$3,501.57	0*
sEVPPI, ($w = 0.5$)	\$3,384.61	\$3,384.61	(\$2,551.99)	\$3,384.61	0*

In these results, a value of 0* indicates that the associated EVI method's policy instructed the DM to not gather information. Over all beliefs, Perseus provides the largest increase in expected value. This is anticipated because Perseus uses the POMDP formalism which explicitly models the sequential nature of the problem. Each belief, however, significantly influences the increase in expected value for all methods. This effect is highly dependent on the problem parameters, causing some beliefs to benefit more than others from gathering information. While this dependence makes it difficult to judge why the expected values of each method fluctuate over the different beliefs, these results demonstrate some of the benefits and drawbacks of each method. First, EVPI and sEVPPI are susceptible to gathering information when it is not valuable to do so, indicated by the negative values in Table 10. These methods can overestimate the value of gathering information because they assume the information gathered is perfect. The addition of the weighting factor helps avoid this in some cases, but this is not universal. For example in belief B3, when the weighting factor is reduced to 0.5, EVPI no longer gathers information and avoids a reduction in expected value. In contrast, the weighting factors chosen for sEVPPI do not sufficiently protect the DM from gathering information when it is not valuable to do so. Overall, sEVPPI and EVPPI, are the least reliable for choosing whether or not to gather information. Even though they sometimes yield a large increase in expected value, the potential wrong decision can be costly. In addition, the choice of weighting factor is somewhat arbitrary, as it is intended to approximate the value of imperfect information by simply scaling down the value of

perfect information. Because of this, EVPI and sEVPPI are not considered in the rest of the expected value analysis.

EVSI does not suffer from these drawbacks but always provides a lower bound on the expected value of gathering information. This is due to the fact that it incorporates the imperfect sources of information but does not consider the sequential nature of the problem. For example, the optimum number of "samples" to gather can be dependent upon the specific observations received, and not constant for all observations. Because of this, the DM must choose how much information to gather at the onset and is not able (i.e., the EVSI method computes the expected value as if the DM is not able) to consider whether or not to gather more information dynamically as the information is received. This causes EVSI to yield a lower bound on the expected value, but can never give a negative value like EVPI and sEVPPI.

Perseus, on the other hand, allows the DM to choose whether or not to gather information at each step. To describe this quantitatively, consider the uniform belief (B1). In this case, the increase in expected value for EVSI is \$3,205.19. The EVSI policy that yields this value instructs the DM to gather information about H_t twice but to not gather information about H_s at all. For the EVSI policy, this choice is fixed no matter what observations the DM receives about H_t. In comparison, the increase in expected value for Perseus is \$4,108.15. This value is higher than EVSI because the POMDP framework allows the DM to make further decisions based on the observations received. In order to compute how many times Perseus instructs the DM to gather information, a basic Monte Carlo simulation is required because the observations are stochastic

(Perseus will instruct the DM differently depending on the observations received). Starting at the uniform belief, each trial in the Monte Carlo simulation uses the Perseus policy to decide whether or not to gather information at each step until a design alternative is chosen. Using a simulation with 100,000 samples, on average, the Perseus policy instructs the DM to gather information about H_t 1.63 times (standard deviation of 0.72) and information about H_s 0.84 times (standard deviation of 0.89). These results demonstrate the extent to which the specific observations received from gathering information influence the choice of whether or not to gather more information (they also give a feel for how many times the DM would be likely to gather information in this specific case study problem). It is for this reason that the Perseus policy has a larger increase in expected value than the EVSI policy.

Although both EVSI and Perseus vary significantly with the initial belief, the maximum increase in expected value for the two methods (both under $5,000) is relatively low for the problem in this case study, which considers a total project value over $1,000,000. While the maximum increase in expected value from choosing Perseus over EVSI is only ~$900 (for belief B1), the increase in expected value is ~20% for B1 and is even as high as ~200% for B3. For this specific problem, these values are likely too small to be of significant concern to the DM, but they suggest that the increase in expected values could be much larger for a different definition of the value function.

To investigate this, the value function defined in Section 4.1 is altered to increase the incentive to reduced mass. Keeping the contract baseline payout the same ($1,000,000), the previous mass incentive rate of $50,000/kg is increased to create two

new cases: $500,000/kg and $1,000,000/kg. This increase reflects a scenario where gathering information can significantly increase the expected value, depending on the associated costs. The costs of gathering information are scaled similar to the mass incentive, yielding $15,000 and $30,000 respectively for costs in the two new cases ($1,500 for the original problem). This keeps the expected value of gathering information from become significantly larger than the costs of gathering information. These updated parameters (all others are kept the same), the problem is solved using the uniform belief (B1) because it previously yielded the largest difference in expected value for the original problem. This is anticipated because the uniform belief essentially represents a state of minimum knowledge about the uncertain parameter; the DM considers all possible values of the uncertain parameter equally likely. The results for using EVSI and Perseus for these problems are shown in Table 11 along with the results for the original problem for comparison.

Table 11. Increase in expected value for various mass incentives and costs of gathering information (mass incentive/information cost).

Method	Expected Value Relative to No Information		
	$50,000/$1,500	**$500,000/$15,000**	**$1,000,000/$30,000**
EVSI	$3,205.19	$43,138.30	$88,008.60
Perseus	$4,108.15	$50,177.70	$102,210.10

As with the previous results, all expected values shown are relative to the expected value of choosing a design alternative without gathering information. These

results demonstrate the dependence of both EVSI and Perseus on the value function of the decision problem. Clearly, as the mass incentive and cost of gathering information increase, the expected value for both methods increases significantly. In addition, the difference between the two methods increases as well, where the difference in the expected value between Perseus and EVSI grows from ~$900 for the original problem to ~$14,000 for the largest dollar incentive to reduce mass ($1,000,000). While these results may not conclusively represent all information-gathering problems, in the context of this case study they suggest the following trend: the larger the anticipated value of gathering information, the more beneficial it is to use Perseus and the POMDP framework. Generally, a DM will not know how valuable it will be to gather information at the onset. However, using their knowledge of the problem, they can at least check to see if there is any value to be gained. For example, using the original problem definition for this case study, the mass incentive of $50,000/kg combined with the minimum mass of 1.5 kg reveals that the maximum difference in project value for a design that minimally meets the mass requirement (a mass of 1.5 kg) and one that has zero mass is only $75,000. This is relatively small compared to the total project value. The DM can only expect to gain a small fraction of that amount as it represents the absolute boundaries of what is possible (e.g., a design of zero mass is clearly unrealistic). However, for the case with a mass incentive of $1,000,000, the maximum difference in project value grows to $1,500,000, suggesting that gathering information could significantly raise the projects expected value. EVPI and sEVPPI could serve as viable methods for providing an initial estimate of the value that could be gained in an

information-gathering problem. Even though these methods assume perfect information, they at least establish bounds to help determine if there is any value to be gained from gathering information: if there is little value in gathering perfect information, the same is likely even less value in gathering imperfect information.

While the previous results demonstrate the increase in expected value, they fail to address the cost of implementing the different methods. Any increase in expected value by using Perseus over EVSI is only beneficial if the additional cost of executing Perseus does not outweigh the increase in expected value. In order to gain insight into this, the solution time is compared between the different methods. Because all of the methods are dependent on problem size, multiple problem sizes are selected for comparison. For now, only the number of states is varied (variation in the number of actions is considered in Section 6). The total number of states considered are 101, 401, and 901 (see Section 6.1 for the corresponding number of discrete intervals of the uncertain parameters). For all problem sizes, the mass incentive of $500,000/kg and information cost of $15,000 are used in order to simulate a problem where gathering information is more valuable. The uniform initial belief is again used for this analysis. The solution time for each of the considered methods is presented in Table 12 for the variation in the number of states.

Table 12. Comparison of solution time for varying number of states (actions held constant) using EVI methods and Perseus.

Method	Solution Time, s		
	101 States	401 States	901 States
EVPI ($w = 1$)	0.005	0.012	0.025
EVSI	0.353	5.144	25.579
sEVPPI ($w = 1$)	0.013	0.016	0.022
Perseus	6,955	27,431	93,065

It is clear from this data that Perseus takes multiple orders of magnitude longer to generate the solution in comparison to any of the EVI methods. Although EVPI and sEVPPI are between two and three orders of magnitude faster than EVSI, all EVI methods were calculated in a reasonable amount of time (all under one minute). Even for the smallest problem size, Perseus takes nearly two hours to generate the solution. Upon increasing the size to 901 states, Perseus requires over an entire day while all EVI methods are completed in under a minute. Although it is possible that the computational complexity of Perseus could be afforded in some engineering projects, these results reveal the need for an improved algorithm for solving information-gathering decision problems using the POMDP framework.

5. IGP: AN IMPROVED PBVI ALGORITHM

PBVI, and particularly the Perseus algorithm, demonstrate the ability to significantly reduce the computational complexity of solving POMDPs, but even this reduction is not enough for large POMDPs, such as those with thousands of states or actions (this is demonstrated in Section 6.1). Using the definition in the preceding section, multiple characteristics of an information-gathering problem can cause the POMDP to grow quickly in size: an increase in the number of uncertain parameters, values of each uncertain parameter, available design alternatives and information-gathering actions. As such, many information-gathering problems can become too computationally expensive to solve, even when using Perseus. However, there are certain characteristics of the information-gathering problem structure that make its representation as a POMDP much simpler than the general POMDP. In other words, information-gathering decision problems represent a small subset of the general POMDP definition. By leveraging these characteristics, Perseus can be significantly improved, allowing for large problems to be solved more quickly; this section defines such an improved algorithm, named Information-Gathering Perseus (IGP). IGP significantly reduces the computational complexity of solving POMDPs for information-gathering decision problems.

It is important to note that by creating IGP for a subset of general POMDPs, the algorithm is no longer valid for the general POMDP. The IGP algorithm presented in this work is only valid for solving POMDPs that represent the information-gathering decision problem. Furthermore, the improvements presented in the following subsection

are defined in terms of the specific information-gathering decision problem defined in Section 2.1.1; however, these can be easily expanded to information-gathering problems that do not fit this definition.

5.1 Algorithmic Improvements to Perseus for Information-Gathering Problems

The IGP algorithm takes the original Perseus algorithm presented in Section 3.2.4, and makes several individual improvements based on the characteristics of the information-gathering decision problem. For clarity, each improvement is presented separately in this section. While each of the improvements presented here are in the context of the Perseus algorithm, they are not unique to Perseus and can be altered for implementation in many other PBVI algorithms.

5.1.1 Reduction of Look-Ahead Alpha Vectors

The first improvement over the Perseus algorithm is based on the reduced complexity of the transition function for information-gathering problems. Recall from Section 3.3 that the only non-zero values of the transition function are:

$$T(s,a,s') = \begin{cases} 1 \text{ if } a = x_j \text{ and } s' = Z^* \\ 1 \text{ if } a = t_k \text{ and } s = s' \end{cases}. \tag{58}$$

Thus, for both types of actions, the transition function is only non-zero for approximately $1/|S|$ of the combinations (s,s'), where $|S|$ is the number of states in the problem. Clearly, the number of non-zero values grows with the size of the state space.

Further, the only non-zero value is a value of one. This characteristic can be used to reduce the computations required at each iteration of the Perseus algorithm.

During each iteration, Perseus must compute the look-ahead $\alpha^{a,o}$ vector for every combination of action, observation, and α vector (see Section 3.2.3):

$$\alpha^{a,o}(s) = \sum_{s' \in S} \alpha(s')\, O(a, s', o) T(s, a, s'). \tag{59}$$

Each vector requires a summation over the entire state space, causing the computing complexity to be $O(|S|^2)$; however, the majority of the transition function is zero over this range and the remaining values are identically one. In fact, by splitting the calculation of this vector by action type, the summation can be completely removed:

$$\alpha^{a,o}(s) = \begin{cases} \alpha(Z^*) O(a, Z^*, o) \text{ if } a = x_j \\ \alpha(s) O(a, s, o) \text{ if } a = t_k \end{cases}. \tag{60}$$

This simplification reduces the computing complexity to $O(|S|)$ for each $\alpha^{a,o}$ vector. Particularly for problems with large state, observation, or action spaces, this reduction in complexity can achieve significant savings simply due to the number of $\alpha^{a,o}$ that must be calculated at each iteration. It is important to note that this reduction in complexity only applies to the first step of each iteration, the backup of one belief point b. Once these are calculated, they can be cached and used for the backup of the rest of the belief points until the iteration is completed. Thus, the $\alpha^{a,o}$ calculation comprises only a fraction of the computational resources required at each iteration. As such, the overall

reduction in computational complexity of the Perseus algorithm depends on the specific problem at hand.

5.1.2 Better Belief Subset

The selection of the belief subset for information-gathering decision problems is another area for improvement in the Perseus algorithm. Most PBVI algorithms create this subset by defining an initial belief and taking a random action to update the belief according to the observation received (the observation received is stochastic based on the observation function). This process is repeated multiple times with the belief recorded after each action and observation combination. Perseus in particular defines the initial belief, unless a specified initial belief is given by the DM, as the uniform belief over the state space: $b(s) = 1/|S|$ for all $s \in S$. It then proceeds to choose an action at random for a specified number of steps (adding the new belief at each step to the belief subset) before resetting the belief to the uniform (or initial) belief and beginning again. Each random set of actions up to belief reset is referred to as a random walk. Belief reset occurs when a user specified number of belief points have been collected, denoted by N. This procedure continues until the desired number of belief points B^* is collected and is shown in Figure 20.

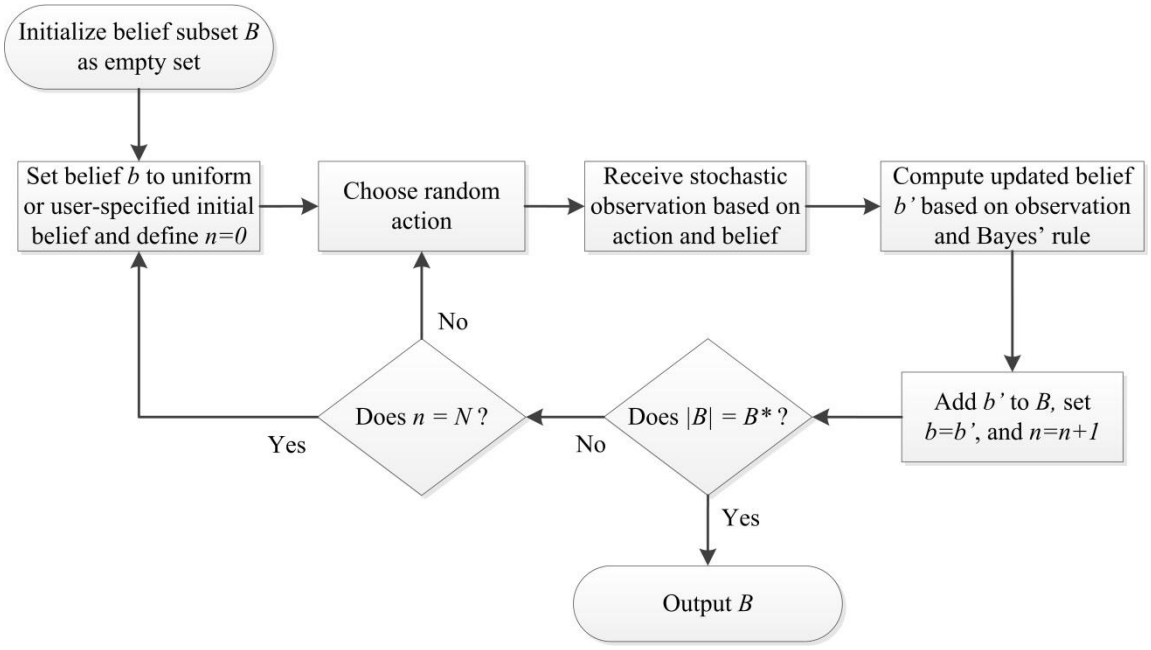

Figure 20. Diagram of Perseus belief subset generation procedure.

There are two aspects of this process that make it inefficient for information-gathering problems. First, in the case of the uniform belief over all of the states, the belief subset chosen will be filled with belief points that can never represent the DM's actual belief. As defined in the transition function, the DM always knows with certainty whether or not a commitment action was chosen. As such, the belief about being in the absorbing state is either 0 or 1 at all times. By initializing the problem with the uniform belief state, the belief of being in the absorbing state is $1/|S|$. Thus, each random walk begins with a belief point that is never feasible for the DM. Further, every other belief point generated in the random walk is also infeasible, except for the terminating belief generated after a commitment action, which sets the belief of being in the absorbing state to one and all other states to zero (this belief is referred to as the absorbing belief). As a

result of this, Perseus solves the POMDP over a large belief space that necessarily does not include the DM's feasible, and much smaller, belief space.

By creating a much more representative belief subset, the computational complexity of solving the POMDP could be decreased significantly due to the fact that fewer beliefs are needed to approximate the smaller space. Additionally, the α vectors associated with the unnecessary beliefs are not needed because they represent the solution to the POMDP outside of the feasible belief space. Thus, the computational complexity can be significantly reduced with a smaller number of beliefs and α vectors involved in each iteration.

The solution to this problem is rather simple: alter the initial belief for the random walks by setting the belief of being in the absorbing state to zero. The initial belief over the rest of the states can be left uniform as before. The initial belief then becomes the following:

$$b_0(s) = \begin{cases} \frac{1}{|S-1|} & \text{for } s \neq Z^* \\ 0 & \text{for } s = Z^* \end{cases}. \tag{61}$$

With this as the initial belief, all subsequent beliefs generated in the random walks are feasible beliefs for the DM. It is also worth noting that the solution based on this representative belief subset is likely to be more accurate. This is due to the fact that the belief subset space used to solve the POMDP is an approximate representation of the feasible belief space of the DM. PBVI uses the belief subset to approximate the policy for beliefs outside of this subset. Thus, the closer the DM's belief is to the belief subset space, the more accurate the associated solution is expected to be. Because the original

Perseus algorithm uses a belief subset with points outside of the DM's feasible space, the solution for beliefs within the DM's feasible space is more approximate. This causes the solution to likely be less accurate.

The second inefficient aspect of generating the belief subset in Perseus is the definition of the random walks: a random action is chosen at each step and the random walk terminates after a finite number of steps. However, due to the nature of the information-gathering problem, the decision terminates once a commitment action is chosen. So, when a commitment action is chosen, the belief is updated to the absorbing belief. If this occurs before a random walk is finished, the belief at all remaining steps will be the same absorbing belief (because the process can never leave the absorbing state), creating many duplicates of the same absorbing belief within the belief subset. As a result, the size of the belief subset would need to be expanded in order to contain the necessary number of *unique* beliefs.

From the previous description, it is clear that once a commitment action is chosen, the beliefs generated at all subsequent steps are redundant. As such, IGP alters the random walk procedure in the following way: restrict the action choices at each step to only the information-gathering actions. This ensures that each random walk reaches termination without adding the absorbing belief. Instead, the absorbing belief is added to the belief subset separately, to insure that it is only added once. Another potential solution to this problem would be to add a termination condition to each random walk such that it terminates when a commitment action is chosen. While this would successfully avoid the creation of duplicate absorbing beliefs, there is one flaw to this

method. In the general information-gathering problem, the number of commitment actions will likely be greater (potentially an order of magnitude or more for some problems) than the number of information-gathering actions. Because of this, at each step of the random walk, it is much more likely for Perseus to choose a commitment action than information-gathering action. By terminating the random walks when a commitment action is chosen, it is unlikely that many random walks will make it past the first or second step. This greatly reduces the exploration of the feasible belief space of the DM, thus making the belief subset less representative. By restricting the actions to the information-gathering actions, the belief space is more adequately explored.

5.1.3 Better Initial Value Function

For any PBVI algorithm, the value function must be initialized to some value before the first iteration. For Perseus, and many other PBVI algorithms, the initial value function must always be a lower bound on the converged solution with each iteration increasing the value function toward the true solution [54]. In order to ensure this, Perseus initializes the value function to a single α vector with all of the components equal to $\frac{1}{1-\gamma}\min_{s,a} R(s,a)$, which is the present value of receiving the lowest possible reward at every step. In general, this is necessary because little or no prior knowledge is available about the solution to the general POMDP, so the lower bound must be established as the worst possible case. For the information-gathering problem however, part of the true solution is known a priori.

If the information-gathering actions were removed from the problem, only commitment actions would remain and the sequential decision becomes a one-step decision. In this case, the expected value of these actions is calculated as shown in Section 2.1.2, but instead considering the Reward function of the POMDP:

$$Ev(a) = \sum_{s \in S} b(s)R(s,a). \tag{62}$$

While the information-gathering problem turns this one-step decision into a sequential decision, the problem always terminates once one of these commitment actions is chosen. Thus, the value of a commitment action only depends on the immediate reward at the current step, and not on the rewards at future steps. (Although in reality the problem terminates with a commitment action, the infinite-horizon POMDP never terminates. To account for this, once a commitment action is chosen, the reward for all future steps is set to zero, thereby imitating the real problem.) Thus, the expected value of choosing the commitment action can always be calculated as a function of the DM's belief. By combining this with the definition of the POMDP value function, the value function α vectors for the commitment actions are defined as follows:

$$Ev(a) = \sum_{s \in S} b(s)\alpha(s) = \sum_{s \in S} b(s)R(s,a). \tag{63}$$

From the preceding equation, $\alpha(s) = R(s,a)$. So, once the information-gathering problem is defined as a POMDP, the true value function α vectors associated with the commitment actions are already known with certainty. This is due to the fact that

information-gathering actions are only chosen if they are expected to increase the value of the commitment action chosen at a later step. The commitment actions are always available to the DM and are only avoided if an information-gathering action is expected to be more valuable. As such, the commitment actions represent the lower bound on the true value function.

Because a lower bound of the converged solution is known before beginning the Perseus algorithm, significant computational expense can potentially be saved by initializing the value function to this lower bound, instead of the arbitrary lower bound prescribed by the Perseus algorithm. This lower bound defines the initial value function as the set $\alpha_{x_j}(s) = R(s, x_j)$ for all $x_j \in A$. Because of this, the initial value function is much closer to the converged solution, likely reducing the number of iterations necessary to get a sufficiently accurate solution. In fact, this value function can potentially be the exact converged solution if gathering information in a particular problem is never expected to add value (for example if the cost of gathering the information is exceedingly high).

This improved initial value function also addresses a potential shortcoming of the POMDP representation. As mentioned in Section 3.2, the infinite-horizon POMDP requires the use of a discount factor $\gamma \epsilon [0,1)$, which for many information-gathering problems is not representative of reality. For these problems, the discount factor would need to be set to 1, or arbitrarily close to 1. For the Perseus algorithm, this is particularly problematic because of the initial value function definition: $\frac{1}{1-\gamma} \min_{s,a} R(s, a)$. For the information-gathering problem, the minimum reward will always be negative (the cost

of gathering information results in a negative reward). Due to this, the initial value function is proportional to $-\frac{1}{1-\gamma}$; the closer γ gets to 1, the farther the initial value function gets from the true solution. Because of the form of the Bellman backup equation, the movement of the value function toward the true solution is not necessarily proportional to the distance from the true solution. As a result, the farther the initial value function is from the true solution, the longer Perseus takes to find a sufficiently accurate solution. In order to represent information-gathering problems with a discount factor arbitrarily close to 1, Perseus becomes prohibitively expensive to execute. IGP, however, does not possess this characteristic because the initialization of the value function is *independen*t of the discount factor.

While improving the initialization of the value function both reduces computational expense and resolves the discount factor limitation, it also accents a particular shortcoming of the Perseus algorithm. Perseus avoids performing a full Bellman backup by backing up individual belief points and checking if the resulting α vector improves other belief points. As noted in Section 3.2.4, this includes the case where $b \cdot \alpha = V_n(b)$ in order to account for the true solution being found for a region of the belief space that includes b. For the general POMDP, it is unlikely that this will occur. For the information-gathering problem with the improved initial value function however, it becomes much more likely. This is because the improved initialization of the value function is guaranteed to contain at least a portion of the true solution. Thus, if a belief point is chosen that lies in the region of the true solution, the backup process will not increase the value of the belief and will produce the exact same α vector.

Because of this phenomenon, it is possible at the conclusion of the first iteration for V_1 to be identicall to V_0 when the true solution has in fact not been found (this is also possible at any other iteration but it best understood by focusing on the first iteration). In order for this to occur, each random belief point chosen during the first iteration must lie in the region of the belief space where the initial value function represents the true solution (note that this does not mean that every point in the belief subset lies in this region, just the random points chosen). In this case, the backup of each belief point adds an identical α vector to V_1 from V_0. Once all α vectors are added to make $V_1 = V_0$, the iteration terminates because the value of every belief point has been "improved" (i.e., $V_1(b) = V_0(b)$ for all $b \in B$).

In general, the chance of this occurring (at any iteration in the process but most likely at the first iteration) is dependent on the extent to which the initial value function represents the true solution; the more representative the initial value function is over the belief space, the more likely a random belief point, chosen from the belief subset, is to reside in this region. As noted previously, the initial value function is identical to the true solution for a particular problem if the DM would never expect it to be valuable to gather information. Thus, the less likely it is to be valuable to gather information, the more likely it is for each random belief point to lie in the region of the true solution. Because of this, it is possible for two successive iterations to yield the same value function if all random belief points are taken from this region, which results in $\epsilon = 0$. This can potentially terminate the algorithm after the first iteration with a solution that indicates gathering information is never valuable, when in fact this may not be the case.

This problem is unique to Perseus and similar algorithms that do not update every belief point independently.

In order to keep this potential false indication of convergence from occurring, IGP monitors the value function evolution to ensure that the value function produces at least one new α vector at each iteration. It does so by maintaining a set B_{rand} that is initialized to $B_{rand} = B$ at the beginning of each iteration. As each random belief point b is backed up, b is removed from B_{rand}. Once the algorithm improves all of the belief points in B (i.e, when $V_{n+1}(b) \geq V_n(b)$ for all $b \in B$), instead of directly moving to the next iteration, IGP adds the following value function monitoring loop:

While $V_{n+1}(b) = V_n(b)$ for all $b \in B$, and B_{rand} is not empty
    Choose random $b \in B_{rand}$ and compute $\alpha = backup(b, V_n)$
    If $\alpha \cdot b \leq V_n(b)$
        Remove b from B_{rand}
    Else
        Add α to V_{n+1} and compute $V_{n+1}(b)$ for all $b \in B$
    EndIf
EndWhile

This extra measure requires that a new α vector be added to V_{n+1} before the next iteration can begin. The only exception is if every belief point $b \in B$ has been backed up and the value function remains unchanged. In this case, the true solution has been found for all $b \in B$.

While this additional monitoring of the value function evolution ensures that a false indication of convergence is not given, it can cause additional computational expense. This is particularly likely in problems where there are very few belief points in B that, when backed up, will create a new α vector (i.e., when it is only valuable to test for a very small region of the belief space). Because of this, it would be beneficial if the belief points that are more likely to generate a new α vector were chosen first during every iteration, in order to avoid entering the loop presented in the previous paragraph. Unfortunately, any method of doing so must, in general, be highly tailored to each specific information-gathering problem.

There is, however, one observation that can be made about these types of problems: the DM's initial belief b_0 is one of the more likely belief points to reside in a region of the belief space for which it is valuable to gather information. This is due to the fact that all of the other belief points in B are generated from random walks that begin at b_0 (see Section 3.2.4). So in most cases, the other belief points likely (but not always) represent a belief with reduced uncertainty. Because these beliefs have reduced uncertainty compared to b_0, it is generally less likely that gathering information is more valuable than for b_0. As such, IGP leverages this and always selects the initial belief of the DM as the first belief to backup in every iteration. While this is not a guarantee to avoid the value function monitoring loop, for many problems it will significantly reduce the likelihood of entering the loop, thereby reducing computational expense.

In order to more visually present the improvements defined in this section and to compare them to Perseus, Figure 21 presents the general procedure for IGP where the changes and additions to Perseus are shown in bold font.

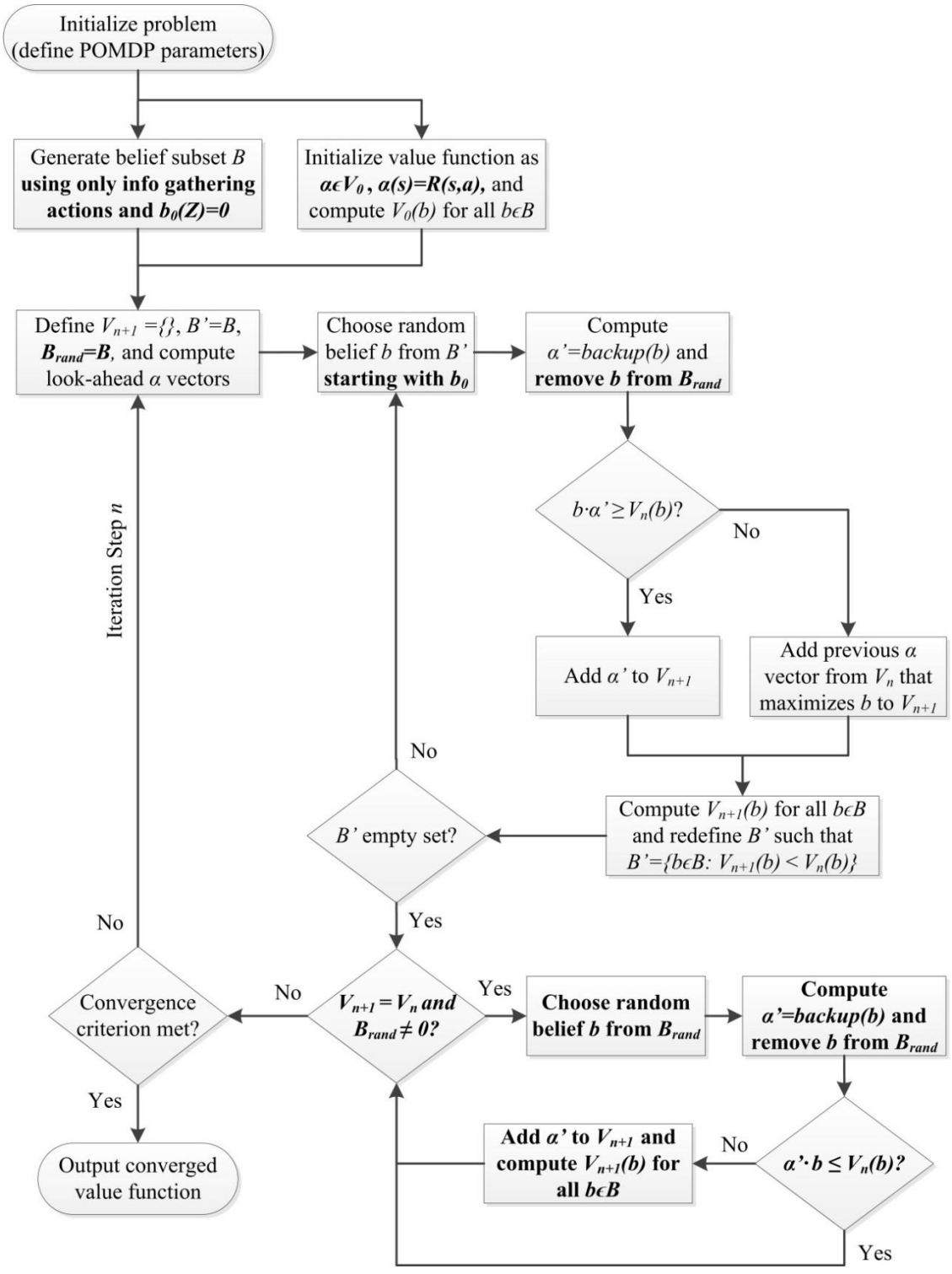

Figure 21. Diagram of IGP algorithmic steps.

6. IGP AND PERSEUS COMPARISON USING CASE STUDY

6.1 Evaluation and Comparison Methods

In order to determine the improvement of the IGP algorithm for solving information-gathering problems when represented as POMDPs, the reduction in the computational complexity, relative to Perseus, must be demonstrated. In order to investigate the impact of problem size on the IGP algorithm improvement over Perseus, the case study is solved at multiple problem sizes. The size of the problem is determined by both the number of states representing the uncertain parameters and the number of commitment actions available to the DM (number of design alternatives to choose from). Both the number of states and the number of actions, which are a function of the discretization of the parameter range (see Section 3.3), are varied independently to investigate the impact of the size of each parameter individually. Table 13 and Table 14 provide the levels of discretization used for both the uncertain parameters and the design variables as well as their associated number of states and actions for the POMDP representation. For the states and actions, the discretization of the variables for the two separate design concepts are held equal ($N_{H_s} = N_{H_t}, N_{L_s} = N_{L_t}$).

Table 13. Parameters for the variation of number of states.

Parameter	Value(s)
Number of Intervals of H_s and H_t ($N_{H_s}$and N_{H_t})	[5,10,15,20,30,40]
Associated Number of States	[26,101,226,401,901,1601]
Number of Intervals of L_s and L_t (N_{L_s} and N_{L_t})	10
Associated Number of Actions	22

Table 14. Parameters for the variation of number of actions.

Parameter	Value(s)
Number of Intervals of H_s and H_t, $N_{H_s}$and N_{H_t}	10
Associated Number of States	101
Number of Intervals of L_s and L_t, N_{L_s} and N_{L_t}	[20,100,200,450,800]
Associated Number of Actions	[22,102,402,902,1602]

To compare each algorithm equally, the same set of algorithm parameters (except for the discount factor) are applied for every problem size in this case study: the number of sample beliefs used to approximate the belief space is 100,000, and the convergence criterion ϵ is set to $1x10^{-6}$. As described in Section 5.1.3, IGP provides the ability to solve problem sizes with a discount factor arbitrarily close to 1, whereas Perseus becomes prohibitively expensive to solve for such a discount factor. To demonstrate this, a small study is performed for a single problem size that demonstrates the drastic increase in solution time of Perseus as the discount factor approaches 1.

In order to account for this, IGP is first compared to Perseus over a smaller range of problem sizes to avoid the prohibitive computational expense of Perseus: only a variation in the number of states is considered using the values presented in Table 13 (where only interval sizes 5, 10, 15, 20 and 30 were computed) with a discount factor of $1 - 1x10^{-3}$. This allows each algorithm to solve the exact same set of problems, providing a fair comparison of computational expense. In addition, this allows for validation that the IGP algorithm produces the same solution as Persues. To perform this validation, the solution from both IGP and Perseus is used to calculate the value $V(b)$ (the expected value of acting optimally as defined in Section 3.1) for a set of beliefs; this is done for every problem size considered For each case, the set of beliefs is chosen randomly from the set of sample beliefs generated by the IGP solution (10,000 beliefs for each problem). These beliefs were chosen as they more accurately represent the possible beliefs of the DM when compared to those generated using Perseus. The value of each belief point as calculated by IGP is compared to the value from Perseus using percent error relative to Perseus to demonstrate the agreement between the solutions.

With this direct comparison completed, the IGP algorithm is then applied over the entire range of problem sizes presented in Table 13 and Table 14 using discount factors of $1 - 1x10^{-3}$ and $1 - 1x10^{-7}$ to demonstrate the minimal increase in solution time when using IGP for a discount factor arbitrarily close to 1.

Using the same software and machine as in Section 4, computational complexity is measured in terms of the wall-clock time required to compute the solution for each algorithm. While wall-clock time is problematic for comparing results between different

machines, it is a reasonable approach for this analysis because each individual algorithmic improvement of IGP affects the computational complexity in different ways. Again, in order to reduce the potential random effects of the system on the wall-clock time as well as the inherent stochastic nature of both IGP and Perseus, each IGP solution is calculated 10 times and averaged (the same is not done for Perseus due to large computational expense compared to IGP, generally an order of magnitude or more). As a further demonstration and baseline comparison to Perseus, one single problem size is chosen and solved 100 times for IGP and 10 times for Perseus.

6.2 IGP and Perseus Comparison Results

The results for the computation time of both IGP and Perseus for the variation in the number of states (using a common discount factor of $1 - 1x10^{-3}$) are presented in Figure 22. In order to better represent the results, because the difference in solution time is so drastic, the data are presented using a "Factor of Improvement" in Figure 23, which is the baseline Perseus solution time divided by the improved algorithm solution time.

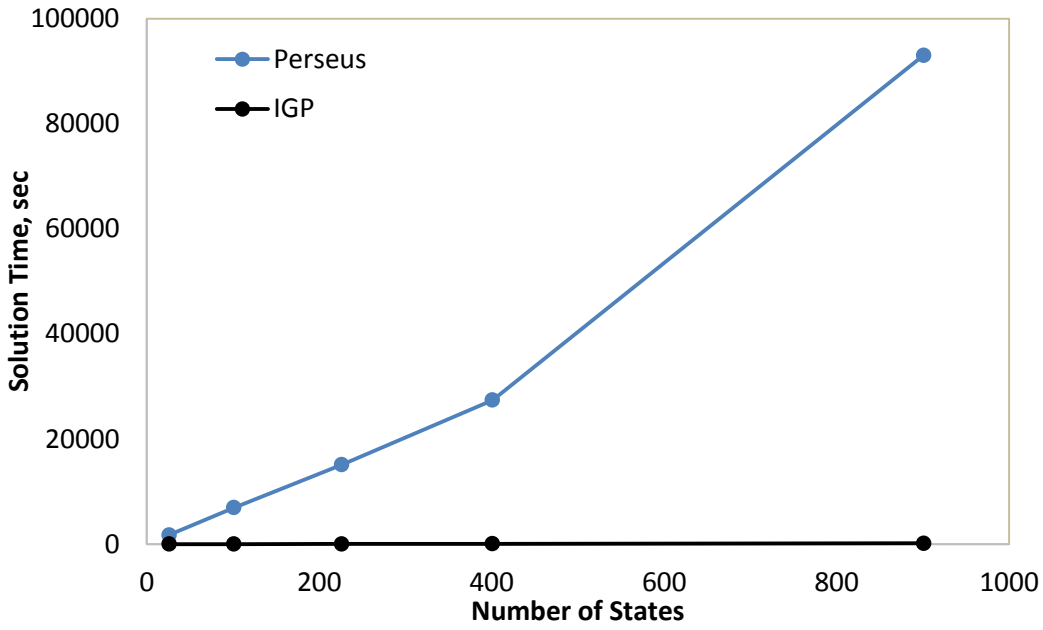

Figure 22. Comparison of IGP and Perseus solution time ($\gamma = 1 - 1x10^{-3}$).

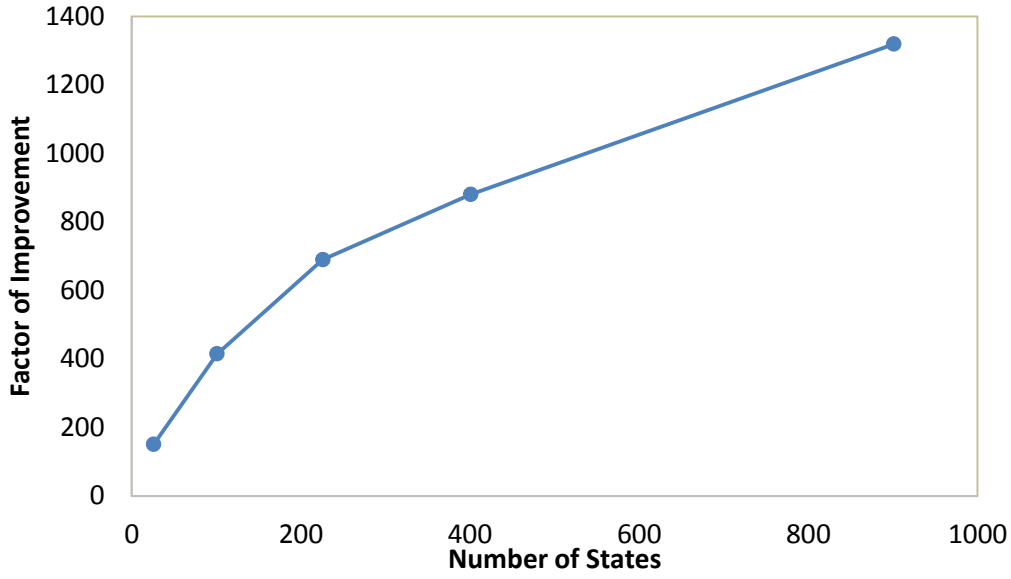

Figure 23. IGP factor of improvement over Perseus ($\gamma = 1 - 1x10^{-3}$).

It is clear in both of these figures that IGP reduces the computational complexity by over two orders of magnitude for even the smallest problem (a more detailed look at the exact solution time of IGP is shown later in this section). In addition, the relative percent error of the expected value of the uniform belief between Perseus and IGP (with Perseus as the reference) is on average ~0.015% for each of the problem sizes considered. This provides a very basic measure of validation that the results of IGP match those of Perseus within a minimal amount of error.

It should be noted that the discount factor of $1 - 1x10^{-3}$ used for comparison between IGP and Perseus was not chosen arbitrarily, but was chosen (through trial and error) as a discount factor that is sufficiently large that the decision problem does not become trivial. For example, a discount factor of $1 - 1x10^{-2}$ applied to this problem causes it to never be valuable to gather information in any of the problem sizes of this case study, regardless of the solution algorithm applied. This is because this discount is applied to values of the design alternatives on the order of $1,000,000; even gathering information once reduces the values by around $10,000. For this specific problem, gathering information is not expected to increase the value by even half of this amount. The discount factor must be sufficiently close to one that this difference is much less than $10,000. In addition, the discount factor was not chosen to arbitrarily close to 1 because of the drastic increase in computation time required by Perseus with the discount factor. This increase is demonstrated in Figure 24, where the discount factor is presented in the form $\gamma^* = -log(1 - \gamma)$ to better show the behavior by giving the

magnitude of the exponent x when using the form $\gamma = 1 - 1x10^{-x}$ (for example $\gamma^* = 2$ for $\gamma = 1 - 1x10^{-2}$).

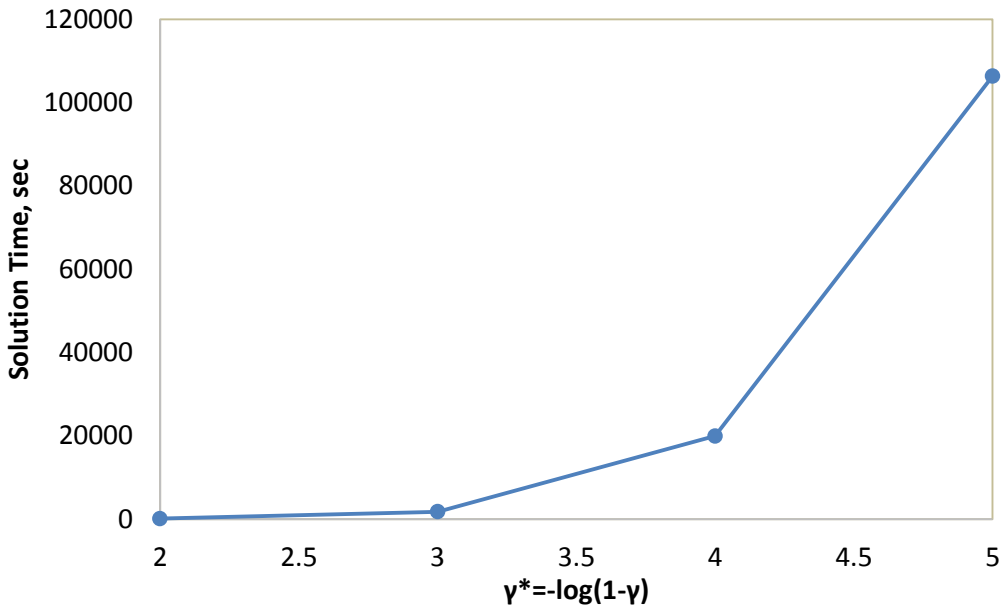

Figure 24. Perseus solution time as a function of increasing discount factor γ closer to 1 using $-log(1-\gamma)$.

For the data in Figure 24, the solution time increases exponentially with the increase in discount factor. Thus, Perseus is not a capable algorithm for solving information-gathering problems in which the value of future actions is not discounted.

The results of the stochastic study are presented in Table 15 with the mean and standard deviation for the 100 trials using IGP and the 10 trials using Perseus. The problem size chosen for this study was 101 states and 22 actions using a discount factor of $1 - 1x10^{-3}$ and all other parameters as defined previously in Section 5.4. The standard deviation for IGP is ~10% of the total solution time while for Perseus the

standard deviation is under 4%. While this is a marginal increase in variation compared to Perseus (2.5 times greater), a variation of 10% in the computation time is relatively insignificant in terms of the reduction in computational complexity provided by IGP. For example, a variation in solution time of IGP up two standard deviations (~25 seconds) is still over two orders of magnitude faster than Perseus.

Table 15. Stochastic study of wall-clock time for Perseus and IGP for the same problem with 101 states and 22 actions.

	Perseus (10 Trials)	**IGP (100 Trials)**
Mean, sec	8027.66	20.11
Standard Deviation, sec	310.23	2.4

Using the problem size definitions in Section 6.1, the solution times for IGP are presented in Figure 25 for the full variation in states and in Figure 26 for the full variation in actions. All problems are solved for two different values of the discount factor: $1 - 1x10^{-3}$ and $1 - 1x10^{-7}$. For Perseus, the computation time increases drastically for even a small increase in the discount factor. These results demonstrate that IGP can handle a discount factor arbitrarily close to 1, with a relatively small increase in computational expense. For the variation in states, the solution time generally doubles with the significant increase in discount factor (or in other words, significant decrease in $1 - \gamma$ of 4 orders of magnitude). This increase in computation time is likely due to the fact that a larger discount factor makes it more likely to gather information (because the values at future decisions are not discounted). This causes the generation of

the solution to be more intensive because it must account for the likelihood of more sequential decisions than in the case where future values are discounted.

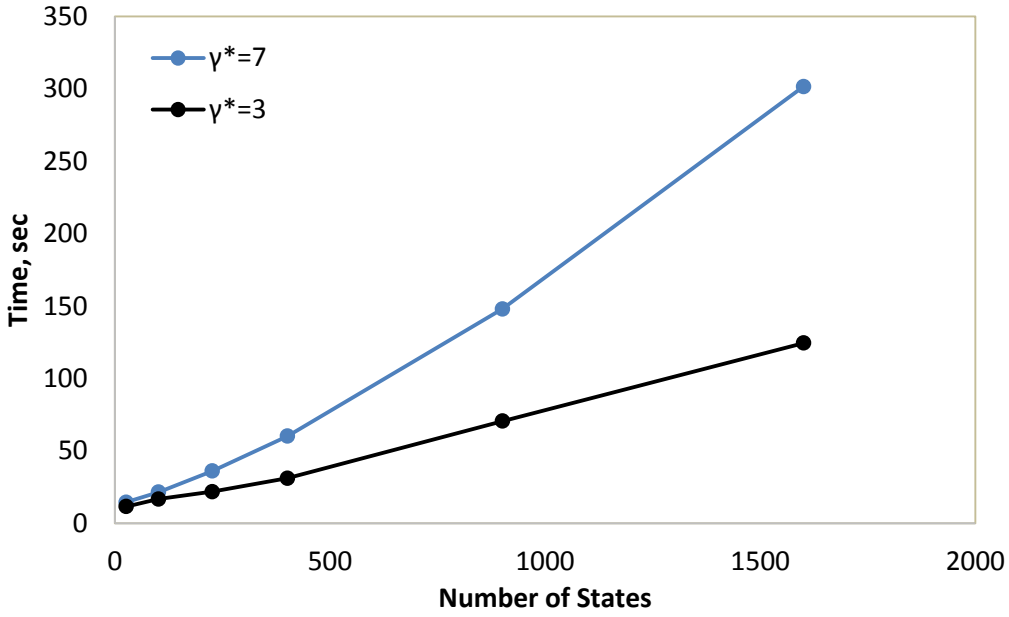

Figure 25. IGP solution time with varying number of states (actions held constant) for $\gamma = 1 - 1x10^{-3}$ and $\gamma = 1 - 1x10^{-7}$.

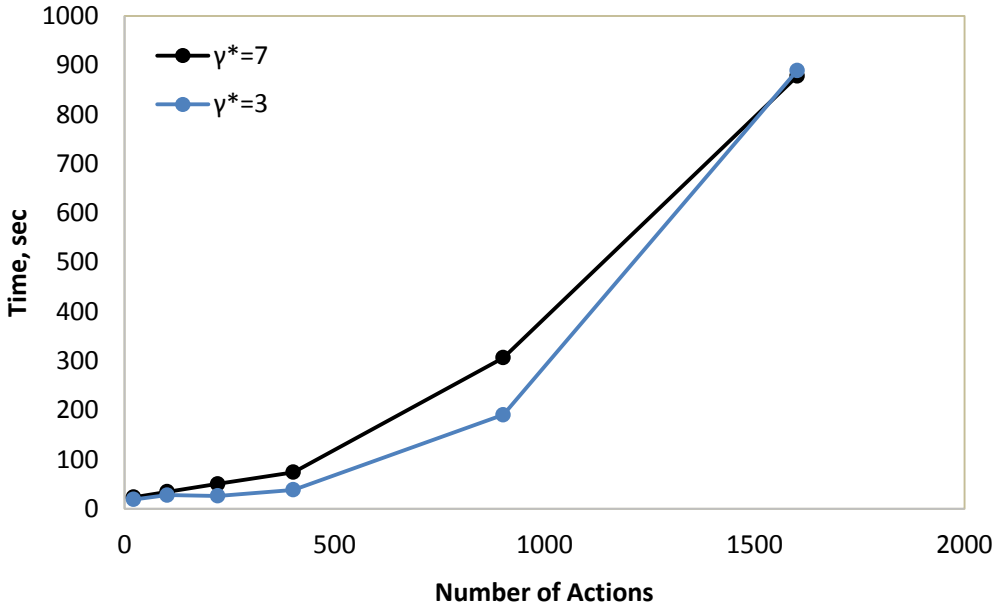

Figure 26. IGP solution time with varying number of actions (states held constant) for $\gamma = 1 - 1x10^{-3}$ and $\gamma = 1 - 1x10^{-7}$.

In general, both the increase in the number of states and the number of actions leads to an increase in required solution time as expected. However, this trend appears to be linear over the state space and exponential over the action space. Further, as the number of actions increases, the difference in solution time between using the two discount factors initially increases but then decreases until the solution time is roughly the same. Upon further examination, this affect appears to be caused by the definition of the initial value function for IGP (see Section 5.1.3). Because the expected value of choosing any design alternative is known a priori, IGP compute the α vector associated with each alternative and combines them to form the initial value function. While there is nothing wrong with this approach, it is possible that this initial value function includes

α vectors for design alternatives that are strictly dominated by other alternatives. Upon examination of the policy solution for all problem sizes in Figure 26, only ~20 design alternatives are included in every policy, regardless of the increase in total alternatives available. This confirms that many of the alternatives (even the majority as the total number of alternatives increases) are strictly dominated. The first iteration of IGP must compute the $\alpha^{a,o}(s)$ for ever α vector (see Section 3.2.3), which includes many α vectors that will be eliminated for all future iterations. This causes the first iteration to take a larger and larger portion of the total solution time as the number of design alternatives increases, to the point where this portion is the majority of the total time. This is the phenomenon that occurs in Figure 26. A more efficient approach to generating the initial value function that eliminates dominated α vectors would likely reduce this effect and should be investigated in future work.

The previous results demonstrate the increase in solution time as the problem size grows, which is directly related to the level of discretization. To compare the value of increasing the level of discretization, Figure 27 and Figure 28 provide a measure of convergence of the solution generated. For each of the problem sizes, for both the varying states and actions, the expected value presented is calculated using the policy for the case of the uniform belief across the uncertain parameters. In both cases, the expected value approaches a constant value as the problem size increases. The slight variation at the tails of the convergence curves is within the tolerance of the convergence criterion chosen for this case study. This provides evidence that information-gathering decision problems with continuous uncertain parameters and design alternatives can be

solved using the POMDP formalism by incorporating the discretization procedure presented in this work. Further, the IGP algorithm provides a reasonable method for solving such problems by reducing the previously prohibitive computational cost of using Perseus.

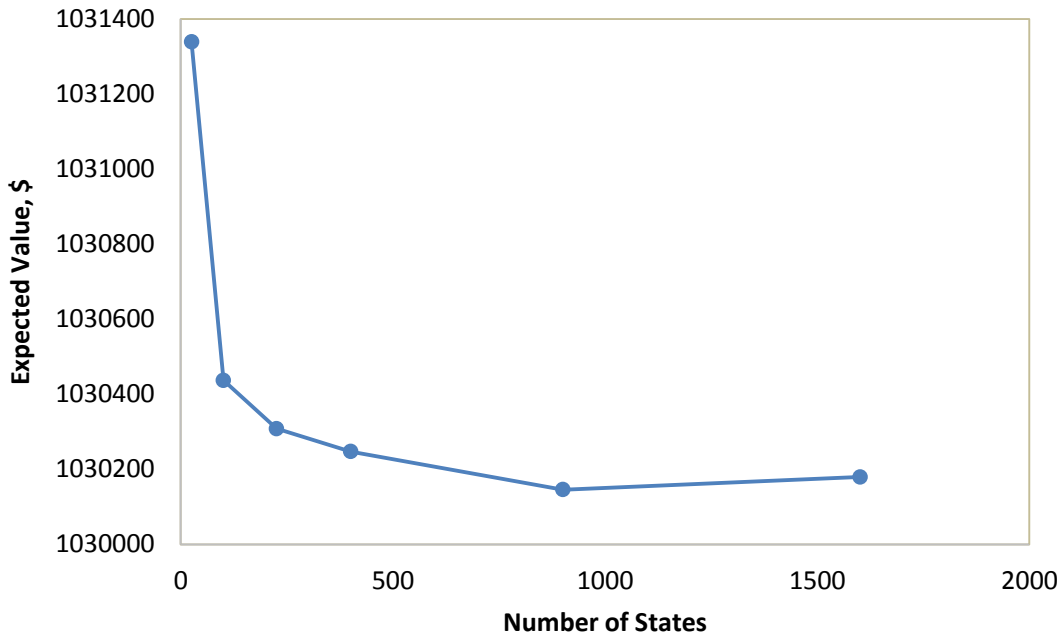

Figure 27. IGP solution for expected value of uniform belief with varying number of states (actions held constant) for $\gamma = 1 - 1x10^{-7}$.

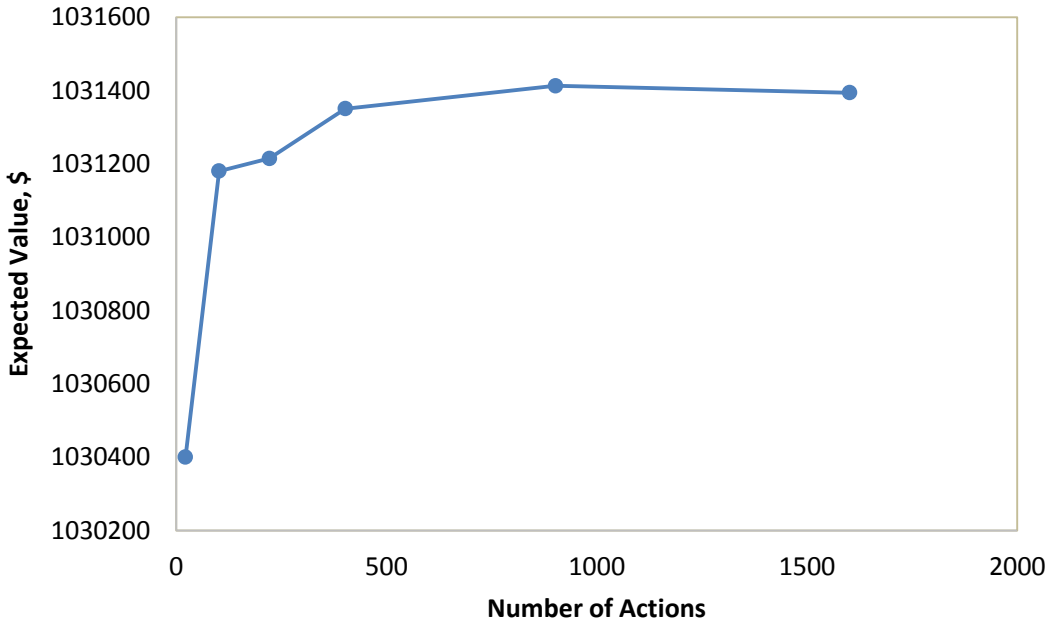

Figure 28. IGP solution for expected value of uniform belief with varying number of actions (states held constant) for $\gamma = 1 - 1x10^{-7}$.

This case study only considers a problem with a limited number of variables (two uncertain parameters, two categories of design variables, and two information-gathering actions). Clearly, an increase in the number of any of these variables would increase the computation time, making this approach questionable for design problems where there are a significant number of design variables or uncertain parameters. However, these results demonstrate that it is in fact a reasonable approach for problems similar to the size of the case study considered. Although this claim is in terms of only the solution time, the following section demonstrates the effectiveness of the approach in leading the DM to make decisions that are expected to be more valuable.

6.3 Comparison of IGP to Other Methods

Having demonstrated the reduced solution time of IGP, consider again the analysis in Section 4.5 where the solution time of Perseus was compared to EVI methods. In this section, Perseus is replaced with IGP in order to compare the improved algorithm to the EVI methods. With the same baseline problem size of 101 states and 22 actions, an independent variation in both the number of states and number of actions is now considered: the total number of states considered are 101, 401, and 901, and the total number of actions considered are 22, 402, and 802 (see Section 6.1 for the corresponding number of discrete intervals of the uncertain parameters and design alternatives). For all problem sizes, the mass incentive of $500,000/kg and information cost of $15,000 are used in order to simulate a problem where gathering information is more valuable. The uniform initial belief is again used for this analysis. The solution time for each of the considered methods is presented in Table 16 for the variation in the number of states and Table 17 for variation in the number of actions.

Table 16. Comparison of solution time for varying number of states (actions held constant) using EVI methods and IGP.

Method	Solution Time		
	101 States	401 States	901 States
EVPI ($w = 1$)	0.005	0.012	0.025
EVSI	0.353	5.144	25.579
sEVPPI ($w = 1$)	0.013	0.016	0.022
IGP	21.54	60.22	148.12

Table 17. Comparison of solution time for varying number of actions (states held constant) using EVI methods and IGP.

Method	Solution Time		
	22 Actions	402 Actions	802 Actions
EVPI ($w = 1$)	0.005	0.003	0.002
EVSI	0.353	0.377	0.399
sEVPPI ($w = 1$)	0.013	0.002	0.005
IGP	26.793	71.732	296.882

The solution times for EVPI and sEVPPI are extremely fast (well under 1 second) for all problem sizes considered. It is for this reason that these methods are often used heuristically; while they are often inaccurate (see Section 4.5), they produce results quickly to provide an initial indication of the value of gathering information. For EVSI, the solution time increases noticeably for the increase in states, but does not increase much with the number of actions. This is reasonable because the number of states is related to the number of "samples" that the DM can receive. Because EVSI considers gathering information multiple times, it must consider every possible combination of samples that can be received. This grows exponentially with the number of states. In contrast, increasing the number of actions only increases the number of design alternatives available after gathering any number of samples. This causes only a relatively minor increase in the solution time.

Considering the comparison of EVSI to IGP, EVSI is two orders of magnitude faster than IGP for the smallest problem. However, by increasing the number of states, EVSI begins to approach the same solution time as IGP. In general, these results suggest

that EVSI can approach similar computational expense as IGP. In such cases, IGP should be used instead of EVSI because of the demonstrated increase in the expected value when using IGP over EVSI. Further, the solution delivered by IGP not only provides the DM with the best action to take at their initial belief, it also dictates the best action to take at each subsequent step based on the information received. These benefits of IGP and the POMDP formalism suggest that IGP should be used whenever it is likely that the computational expense of EVSI and IGP are similar. It is important to note that all of the EVI methods and the POMDP formalism require the same knowledge about the decision problem (values of actions based on uncertain parameters, accuracy of testing action, etc.). All of the parameters that must be known in order to define the appropriate POMDP must also be known to use any of the EVI methods.

The results in this section provide a glimpse of the tradeoff between the expected value and computational cost. This tradeoff is highly dependent on the information-gathering decision problem size, parameters and value function as demonstrated in Section 4.5. In order to make more definitive statements about when to use IGP and the POMDP formalism for solving these problems, a much more in-depth study across a large range of problem sizes, parameters and value functions is necessary. Nevertheless, the results of this case study provide an initial demonstration of the benefit of using IGP over existing EVI methods.

7. CONCLUSION AND FUTURE WORK

The IGP algorithm is a significant improvement over Perseus for solving information-gathering problems. The results presented in Section 6.2 demonstrate both the reduction in computational expense as well as the ability to handle a more realistic discount factor. IGP makes many information-gathering decision problems realistic to solve using the POMDP formalism which were intractable using Perseus. This thesis also shows that the POMDP formalism is a viable method for solving information-gathering decision problems with continuous parameters. The discretization process allows continuous parameters to be represented in a discrete parameter POMDP formulation. Although this process can lead to the creation of a POMDP with a large number of states and action, IGP makes these problems computationally manageable to solve.

The results in Section 4.5 show that the POMDP formalism always leads the DM to make decisions that have a higher expected value than the other methods considered in this work. Section 6.3 also shows that the POMDP formalism can be a viable method for solving information-gathering decision problems even when considering the associated increase in computational expense due to the IGP algorithm. While the results are indeterminate in their extension to all problems, they demonstrate that IGP and the POMDP formalism are of value to a DM, at least for problems similar to those defined in the case study. These results also suggest that this value extends beyond only the specific design problem considered in the case study.

While this thesis revealed the benefit of using the POMDP formalism for solving information-gathering problems with certain properties, the approach is not without its limitations. Most importantly, this method does not currently scale to problems with a large number of uncertain parameters and design alternatives. The problem size increases exponentially with the number of these parameters. Unfortunately, this is not the only factor that can cause the problem size to increase. The specific type of information-gathering decision problem discussed in this thesis neglects information-gathering actions that are deterministic. Deterministic information-gathering actions are those that are not stochastic, and will produce the exact same results each time it is executed (e.g., non-stochastic engineering simulations). This is an issue for the POMDP formalism because a POMDP is memoryless: it does not know what has happened at previous steps in the process, so it does not know whether or not the DM has already performed a deterministic information-gathering action.

In order to account for this, the number of states must be expanded to include the history within the state of the problem. In a way, the presence of a deterministic action creates multiple true states of the problem: the true state of the uncertain variable as it is treated in this work, and the states that represent whether or not the deterministic action has been executed. Because the POMDP formalism only allows a single state of the process at any time, the state space must be expanded to account for every combination of the multiple states. For example, consider the simple case where there are two information-gathering actions available and one is deterministic. In this case, the state space must be doubled: for every true state of the uncertain parameters, there must be

one version of the state for which the deterministic action has been executed and one for which it has not. The deterministic action provides information about the uncertain parameters only if it has not been previously executed, otherwise it provides no new information but the DM still incurs the cost.

While the IGP algorithm is a significant improvement over Perseus, there is still potential to address the limitations discussed. Particularly, future work should investigate the creation of a new decision process formalism. This formalism could be an expansion of a POMDP such that the decision process can have multiple current states. The transition between the multiple states would be governed similar to the reduced POMDP discussed in Section 3.3. For example, consider the problem described in the case study. The new formalism would include one current state for each uncertain parameter. This would reduce the state space from 101 states to 21 states (10 for possible states for each of the two uncertain parameters and one absorbing state). This reduction is the size of the state space grows exponentially with the addition of more uncertain parameters. Consider the case of 10 uncertain parameters; the states space would reduce from 10^{10} states to only 100 states. Investigation into such a new formalism, while not conclusive at this point, could yield an even more efficient algorithm than IGP, because the improvements described in Section 5 would be directly applicable with only minor changes to address the multiple current states.

This thesis completed the necessary first step of demonstrating that using the POMDP formalism can be more valuable than other methods for solving information-gathering decision problems. It also created an algorithm for solving these problems

more efficiently. Future work should expand upon the method as presented in this thesis and consider a more expansive study of information-gathering decision problem parameters to more accurately determine when IGP and the POMDP formalism might best be used.

REFERENCES

[1] Der Kiureghian, A., and Ditlevsen, O., 2009, "Aleatory or Epistemic? Does It Matter?," Structural Safety, 31(2), pp. 105-112.

[2] Malak, R. J., Aughenbaugh, J. M., and Paredis, C. J., 2009, "Multi-Attribute Utility Analysis in Set-Based Conceptual Design," Computer-Aided Design, 41(3), pp. 214-227.

[3] Haimes, Y. Y., 2005, *Risk Modeling, Assessment, and Management*, John Wiley & Sons, Hoboken, New Jersey.

[4] Edwards, W., Miles Jr, R. F., and Von Winterfeldt, D., 2007, *Advances in Decision Analysis: From Foundations to Applications*, Cambridge University Press, New York, New York.

[5] Lee, E., Park, Y., and Shin, J. G., 2009, "Large Engineering Project Risk Management Using a Bayesian Belief Network," Expert Systems with Applications, 36(3), pp. 5880-5887.

[6] Williams, T., 1995, "A Classified Bibliography of Recent Research Relating to Project Risk Management," European Journal of Operational Research, 85(1), pp. 18-38.

[7] Agarwal, H., 2004, "Reliability Based Design Optimization: Formulations and Methodologies," Ph.D. thesis, University of Notre Dame, Notre Dame, Indiana.

[8] Tu, J., Choi, K. K., and Park, Y. H., 1999, "A New Study on Reliability-Based Design Optimization," Journal of Mechanical Design, 121(4), pp. 557-564.

[9] Phadke, M. S., 1995, *Quality Engineering Using Robust Design*, Prentice Hall PTR, Englewood Cliffs, New Jersey.

[10] Taguchi, G., Chowdhury, S., and Taguchi, S., 2000, *Robust Engineering*, McGraw-Hill Professional, New York, New York.

[11] Wu, Y., and Wu, A., 2000, *Taguchi Methods for Robust Design*, American Society of Mechanical Engineers, New York, New York.

[12] Nembhard, H. B., and Aktan, M., 2009, *Real Options in Engineering Design, Operations, and Management*, CRC Press, Boca Raton, Florida.

[13] De Neufville, R., and Scholtes, S., 2011, *Flexibility in Engineering Design*, MIT Press, Cambridge, Massachusetts.

[14] Saleh, J. H., Hastings, D. E., and Newman, D. J., 2001, "Extracting the Essence of Flexibility in System Design," eds., Long Beach, California, pp. 59-72.

[15] Liker, J. K., Sobek, D. K., Ward, A. C., and Cristiano, J. J., 1996, "Involving Suppliers in Product Development in the United States and Japan: Evidence for Set-Based Concurrent Engineering," Engineering Management, IEEE Transactions on, 43(2), pp. 165-178.

[16] Ward, A., Liker, J. K., Cristiano, J. J., and Sobek Ii, D. K., 1995, "The Second Toyota Paradox: How Delaying Decisions Can Make Better Cars Faster," Sloan management review, 36(3), pp. 43-61.

[17] Clemen, R. T., 1996, *Making Hard Decisions: An Introduction to Decision Analysis*, *Pws-Kent*, Boston, Massachusetts.

[18] Raiffa, H., 1974, *Applied Statistical Decision Theory*, Harvard Business School, Boston, Massachusetts.

[19] Hazelrigg, G. A., 1998, "A Framework for Decision-Based Engineering Design," Journal of Mechanical Design, 120(4), pp. 653-658.

[20] Hazelrigg, G. A., 2012, *Fundamentals of Decision Making for Engineering Design and Systems Engineering*, Self-Published.

[21] Krishnamurty, S., 2006, "Normative Decision Analysis in Engineering Design," Decision Making in Engineering Design, 4(4), pp. 21-33.

[22] Lewis, K. E., Chen, W., and Schmidt, L. C., 2006, *Decision Making in Engineering Design*, ASME Press, New York, New York.

[23] Thompson, S. C., and Paredis, C. J., 2010, "An Investigation into the Decision Analysis of Design Process Decisions," Journal of Mechanical Design, 132(12), pp. 121009.

[24] Pratt, J. W., Raiffa, H., and Schlaifer, R., 1995, *Introduction to Statistical Decision Theory*, MIT press, Cambridge, Massachusetts.

[25] Howard, R. A., 1966, "Information Value Theory," Systems Science and Cybernetics, IEEE Transactions on, 2(1), pp. 22-26.

[26] Bradley, S., and Agogino, A. M., 1994, "An Intelligent Real Time Design Methodology for Component Selection: An Approach to Managing Uncertainty," Journal of Mechanical Design, 116(4), pp. 980-988.

[27] Radhakrishnan, R., and Mcadams, D. A., 2005, "A Methodology for Model Selection in Engineering Design," Journal of Mechanical Design, 127(3), pp. 378-387.

[28] Wood, W. H., and Agogino, A. M., 2005, "Decision-Based Conceptual Design: Modeling and Navigating Heterogeneous Design Spaces," Journal of Mechanical Design, 127(1), pp. 2-11.

[29] Panchal, J. H., Paredis, C. J., Allen, J. K., and Mistree, F., 2008, "A Value-of-Information Based Approach to Simulation Model Refinement," Engineering Optimization, 40(3), pp. 223-251.

[30] Messer, M., Panchal, J. H., Krishnamurthy, V., Klein, B., Yoder, P. D., Allen, J. K., and Mistree, F., 2010, "Model Selection under Limited Information Using a Value-of-Information-Based Indicator," Journal of Mechanical Design, 132(12), pp. 121008.

[31] Takai, S., 2010, "A Use of a Mathematical Model in Updating Concept Selection," Journal of Mechanical Design, 132(10), pp. 101009.

[32] Thurston, D. L., and Nogal, A., 2001, "Meta-Level Strategies for Reformulation of Evaluation Function During Iterative Design," Journal of Engineering Design, 12(2), pp. 93-115.

[33] Parmigiani, G., and Inoue, L., 2009, *Decision Theory: Principles and Approaches*, John Wiley & Sons, West Sussex, United Kingdom.

[34] Eckermann, S., and Willan, A. R., 2007, "Expected Value of Information and Decision Making in Hta," Health economics, 16(2), pp. 195-209.

[35] Willan, A. R., 2007, "Clinical Decision Making and the Expected Value of Information," Clinical Trials, 4(3), pp. 279-285.

[36] Willan, A. R., and Eckermann, S., 2010, "Optimal Clinical Trial Design Using Value of Information Methods with Imperfect Implementation," Health economics, 19(5), pp. 549-561.

[37] Ades, A., Lu, G., and Claxton, K., 2004, "Expected Value of Sample Information Calculations in Medical Decision Modeling," Medical Decision Making, 24(2), pp. 207-227.

[38] Shewry, M. C., and Wynn, H. P., 1987, "Maximum Entropy Sampling," Journal of Applied Statistics, 14(2), pp. 165-170.

[39] Shannon, C. E., 2001, "A Mathematical Theory of Communication," ACM SIGMOBILE Mobile Computing and Communications Review, 5(1), pp. 3-55.

[40] Frazier, P. I., Powell, W. B., and Dayanik, S., 2008, "A Knowledge-Gradient Policy for Sequential Information Collection," SIAM Journal on Control and Optimization, 47(5), pp. 2410-2439.

[41] Chaloner, K., and Verdinelli, I., 1995, "Bayesian Experimental Design: A Review," Statistical Science, 10(3), pp. 273-304.

[42] Huan, X., and Marzouk, Y. M., 2013, "Simulation-Based Optimal Bayesian Experimental Design for Nonlinear Systems," Journal of Computational Physics, 232(1), pp. 288-317.

[43] Pertile, P., Forster, M., and Torre, D. L., 2014, "Optimal Bayesian Sequential Sampling Rules for the Economic Evaluation of Health Technologies," Journal of the Royal Statistical Society: Series A (Statistics in Society), 177(2), pp. 419-438.

[44] Plant, R. E., and Wilson, L., 1985, "A Bayesian Method for Sequential Sampling and Forecasting in Agricultural Pest Management," Biometrics, 4(1), pp. 203-214.

[45] Zirbel, S. A., Lang, R. J., Thomson, M. W., Sigel, D. A., Walkemeyer, P. E., Trease, B. P., Magleby, S. P., and Howell, L. L., 2013, "Accommodating Thickness in Origami-Based Deployable Arrays," Journal of Mechanical Design, 135(11), pp. 111005.

[46] Christiansen, B., 2013, Byu Engineers Use Origami for Space Projects, March 1, 2015, http://www.heraldextra.com/news/local/byu-engineers-use-origami-for-space-projects/article_596190a9-9123-51dd-be61-024faf5d4d2f.html.

[47] Hsiao, C., and Malak, R., 2014, "Modeling Information Gathering Decisions in Systems Engineering Projects," eds., Buffalo, New York, pp. 1-12.

[48] Kaelbling, L. P., Littman, M. L., and Cassandra, A. R., 1998, "Planning and Acting in Partially Observable Stochastic Domains," Artificial intelligence, 101(1), pp. 99-134.

[49] Koenig, S., and Simmons, R., 1998, "Xavier: A Robot Navigation Architecture Based on Partially Observable Markov Decision Process Models," Artificial Intelligence Based Mobile Robotics: Case Studies of Successful Robot Systems, pp. 91-122.

[50] Littman, M. L., Cassandra, A. R., and Kaelbling, L. P., 1995, "Learning Policies for Partially Observable Environments: Scaling Up," eds., Tahoe City, California, 95, pp. 362-370.

[51] Hauskrecht, M., and Fraser, H., 2000, "Planning Treatment of Ischemic Heart Disease with Partially Observable Markov Decision Processes," Artificial Intelligence in Medicine, 18(3), pp. 221-244.

[52] Braziunas, D., 2003, "Pomdp Solution Methods," University of Toronto, Tech.

[53] Murphy, K. P., 2000, "A Survey of Pomdp Solution Techniques."

[54] Shani, G., Pineau, J., and Kaplow, R., 2013, "A Survey of Point-Based Pomdp Solvers," Autonomous Agents and Multi-Agent Systems, 27(1), pp. 1-51.

[55] Collopy, P. D., and Hollingsworth, P. M., 2011, "Value-Driven Design," Journal of Aircraft, 48(3), pp. 749-759.

[56] Brown, O. C., Eremenko, P., and Collopy, P. D., 2009, "Value-Centric Design Methodologies for Fractionated Spacecraft: Progress Summary from Phase 1 of the Darpa System F6 Program," eds., Pasedena, California, pp. 2009-6540.

[57] Castagne, S., Curran, R., and Collopy, P., 2009, "Implementation of Value-Driven Optimisation for the Design of Aircraft Fuselage Panels," International journal of production economics, 117(2), pp. 381-388.

[58] Cheung, J., Scanlan, J., Wong, J., Forrester, J., Eres, H., Collopy, P., Hollingsworth, P., Wiseall, S., and Briceno, S., 2012, "Application of Value-Driven Design to Commercial Aeroengine Systems," Journal of Aircraft, 49(3), pp. 688-702.

[59] Barto, A. G., Sutton, R. S., and Watkins, C. J., 1990, "Sequential Decision Problems and Neural Networks," eds., pp. 686-693.

[60] Sigaud, O., and Buffet, O., 2013, *Markov Decision Processes in Artificial Intelligence*, John Wiley & Sons, Hoboken, New Jersey.

[61] Puterman, M. L., 2009, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, John Wiley & Sons, Hoboken, New Jersey.

[62] Papadimitriou, C. H., and Tsitsiklis, J. N., 1987, "The Complexity of Markov Decision Processes," Mathematics of operations research, 12(3), pp. 441-450.

[63] Madani, O., Hanks, S., and Condon, A., 1999, "On the Undecidability of Probabilistic Planning and Infinite-Horizon Partially Observable Markov Decision Problems," eds., Orlando, Florida, pp. 541-548.

[64] Spaan, M. T., and Vlassis, N. A., 2005, "Perseus: Randomized Point-Based Value Iteration for Pomdps," Journal of Artificial Intelligence Research, 24(1), pp. 195-220.

[65] Erez, T., and Smart, W. D., 2012, "A Scalable Method for Solving High-Dimensional Continuous Pomdps Using Local Approximation," eds., Catalina Island, California, pp. 160-167.

[66] Nicol, S., and Chadès, I., 2012, "Which States Matter? An Application of an Intelligent Discretization Method to Solve a Continuous Pomdp in Conservation Biology," PloS one, 7(2), pp. e28993.

[67] Porta, J. M., Vlassis, N., Spaan, M. T., and Poupart, P., 2006, "Point-Based Value Iteration for Continuous Pomdps," The Journal of Machine Learning Research, 7(1), pp. 2329-2367.

[68] Lagoudas, D. C., 2008, *Shape Memory Alloys: Modeling and Engineering Applications*, Springer, New York, New York.

[69] Hartl, D. J., Solomou, A., Lagoudas, D. C., and Saravanos, D., 2012, "Phenomenological Modeling of Induced Transformation Anisotropy in Shape Memory Alloy Actuators," eds., San Diego, California, pp. 83421M-83421M-14.

[70] Lagoudas, D., Hartl, D., Chemisky, Y., Machado, L., and Popov, P., 2012, "Constitutive Model for the Numerical Analysis of Phase Transformation in Polycrystalline Shape Memory Alloys," International Journal of Plasticity, 32(33), pp. 155-183.

[71] Griffin, S., Welton, N. J., and Claxton, K., 2009, "Exploring the Research Decision Space: The Expected Value of Information for Sequential Research Designs," Medical Decision Making, 30(2), pp. 155-162.

[72] Reddy, J. N., 2004, *Mechanics of Laminated Composite Plates and Shells: Theory and Analysis*, CRC press, Boca Raton, Florida.

[73] Tabesh, M., Lester, B., Hartl, D., and Lagoudas, D., 2012, "Influence of the Latent Heat of Transformation and Thermomechanical Coupling on the Performance of Shape Memory Alloy Actuators," eds., Stone Mountain, Georgia, pp. 237-248.

APPENDIX

9.1 Bending of the SMA Sheet Concept

In order to predict the unfolding behavior of the two concepts (for any choice of design alternative), the firm has chosen to use pure bending and pure torsion analysis for the SMA sheet and SMA torque tube respectively, assuming a constant applied moment that opposes the unfolding due to the stiffness of the non-actuated connections between panels in the solar array (while in general more detailed and rigorous analysis methods would likely be used in practice, the basic analysis presented here is sufficient for this case study).

First, consider the SMA sheet as a beam in pure bending. The beam is subject to planar motion, where the axial direction is denoted by x and the direction of the beam normal is denoted by z. Further, by assuming unshearable deformation in the beam, the shear strain γ_{xz} reduces to zero. For a beam with a curved reference configuration along the axial direction, the total strain is given as follows [72]:

$$\varepsilon_{xx} = \varepsilon_{xx}^{0} + z\varepsilon_{xx}^{1}, \tag{A.1}$$

$$\varepsilon_{xx}^{0} = \frac{du}{dx} + \frac{w}{R_1} + \frac{1}{2}\left(\frac{dw}{dx} - \frac{u}{R_1}\right)^2, \tag{A.2}$$

$$\varepsilon_{xx}^{1} = -\frac{d^2w}{dx^2}, \tag{A.3}$$

where u and w are the axial and normal displacements of the centroidal axis respectively, and R_1 is the radius of curvature of the beam in the reference configuration.

Assuming that the Young's modulus E remains constant throughout SMA transformation and that the only source of inelastic strain is the transformation strain of the SMA, the stress along the axial direction in the beam is given by the following:

$$\sigma_{xx} = E(\varepsilon_{xx}^0 + z\varepsilon_{xx}^1 - \varepsilon^t + \varepsilon^{t0}), \tag{A.4}$$

where ε^t is the transformation strain and ε^{t0} is the initial pre-strain. Under the assumption that the beam thickness t_s is small compared to R_1, the following relation for the moment load is obtained (the axial load is omitted as it is not needed for this analysis):

$$M = L_s \int_{-\frac{t_s}{2}}^{\frac{t_s}{2}} \sigma_{xx} z dz = \frac{L_s E t_s^3 \varepsilon_{xx}^1}{12} - M^t, \tag{A.5}$$

$$M^t = L_s \int_{-\frac{t_s}{2}}^{\frac{t_s}{2}} E(\varepsilon^t - \varepsilon^{t0}) z dz, \tag{A.6}$$

where L_s is the width of the beam (which in this case is the length of the SMA sheet as defined previously). Here, the displacements are assumed to be small in comparison to R_1 resulting in

$$\frac{u}{R_1} \approx 0, \qquad \frac{w}{R_1} \approx 0. \tag{A.7}$$

By assuming symmetry at the location of analysis, $\frac{dw}{dx} = 0$. Incorporating all of the previous assumptions, Equation 5 reduces to the following:

$$M = -\frac{L_s E t_s^3}{12}\frac{d^2 w}{dx^2} - M^t. \quad (A.8)$$

Linear pre-strain through the thickness is assumed. With no other inelastic strains, the SMA provides the maximum attainable transformation strain H_s, which is reached at the top and bottom faces of the beam. This results in the following equation:

$$\varepsilon^{t0} = -\frac{2zH_s}{t_s}. \quad (A.9)$$

The transformation strain is a function of the martensitic volume fraction ξ as follows:

$$\varepsilon^t = \xi\varepsilon^{t0} = -\frac{\xi 2zH_s}{t_s}. \quad (A.10)$$

By incorporating the transformation strain fields into M_t in Equation 6, Equation 5 reduces to the following:

$$M = -\frac{L_s E t_s^3}{12}\frac{d^2 w}{dx^2} + \frac{L_s E H_s(\xi - 1)t_s^2}{6}. \quad (A.11)$$

The radius of curvature of the beam R and the moment of inertia of the cross-section I are defined:

$$\frac{d^2 w}{dx^2} = \frac{1}{R} - \frac{1}{R_1}, \quad (A.12)$$

$$I = \frac{L_s t_s^3}{12}. \quad (A.13)$$

Substituting Equations 12 and 13 into Equation 11 and rearranging the terms yields the expression:

$$\frac{1}{R} - \frac{1}{R_1} = -\frac{M}{EI} + \frac{L_s H_s t_s^2 (\xi - 1)}{6I}. \qquad (A.14)$$

This analysis requires that both the initial configuration be defined and that the pre-strain be equivalent to the maximum transformation strain at the top and bottom faces of the beam. However, this analysis must be adapted to the specific application of the SMA sheet for the solar array in this case study, because both the initial configuration and the pre-strain are dependent upon the design alternative chosen (L_s). First, the initial curved configuration must be determined. For this application, the SMA sheet should unfold from the stowed configuration to the flat configuration under the opposition of a constant applied moment M_0, which is included in this analysis to represent the moment that opposes the unfolding of the SMA sheet due to the stiffness of the non-actuated connections between panels in the solar array (while in general more detailed and rigorous analysis methods would likely be used in practice, the basic analysis presented here is sufficient for this case study). As such, once the SMA sheet has transformed to austenite and all transformation strain has been recovered, it must be perfectly flat even with the opposing moment still present. Because of this, the initial radius of curvature of the sheet R_0 (i.e., the curvature of the stress-free sheet before any pre-strain has been generated) must be defined such that with the application of M_0, the sheet reaches the flat configuration. This is demonstrated in Figure 29.

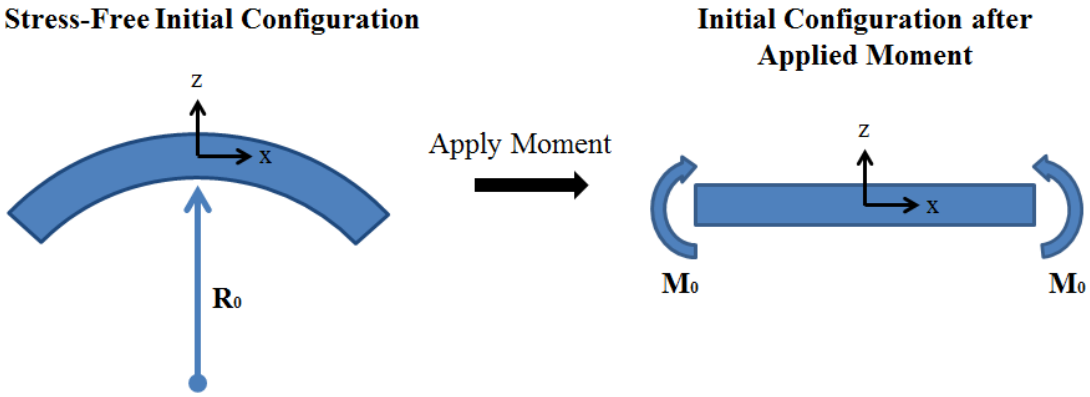

Figure 29. Initial configuration of the SMA sheet with and without applied moment.

To solve for R_0, only elastic strain is considered ($\varepsilon^t = \varepsilon^{t0} = 0$). Because the sheet bends to flat under the application of M_0, the final radius of curvature is infinite, yielding $\frac{1}{R} = 0$ and reducing Equation 12 to the following:

$$\frac{d^2w}{dx^2} = -\frac{1}{R_0}. \qquad (A.15)$$

Substituting Equation 15 into Equation 8, considering only elastic strain, and rearranging terms results in the following expression:

$$R_0 = \frac{L_s E t_s^3}{12 M_0}. \qquad (A.16)$$

This determines the initial configuration of the SMA sheet without any pre-strain. The SMA sheet must be deformed to reach the stowed configuration. The stowed configuration is defined by the stowed radius of curvature R_s. Further, the sheet must

reach the stowed configuration in a stress-free state. Because of this, all pre-strain generated must be inelastic. If H_s is sufficiently large, all of the inelastic strain will be transformation strain. However, if H_s is too small, plastic strain must be generated in the SMA by bending the sheet to a tighter radius of curvature than R_s and yielding the SMA. In this case, the SMA will not recover the plastic strain upon transformation to austenite, causing the sheet to not unfold completely flat. For any set of sheet parameters, it must be determined whether or not the sheet can unfold completely flat, and if it cannot, the final radius of curvature must be calculated.

To do so, the application of M_0 must be more thoroughly defined. In reality, the moment opposing the sheet is only applied once the sheet begins to transform while in the stowed configuration; as the sheet is heated and begins to recover transformation strain, it will initially not unfold at all because the opposing moment will grow as the transformation strain is recovered. However, once the moment grows to reach M_0, the transformation strain has fully overcome the moment and the sheet will begin to unfold.

While not identical to the previous description, this is similar to the following situation. Consider the stowed configuration (radius of curvature R_s) and apply M_0 *entirely* before heating. This will cause the sheet to fold tighter than the stowed configuration through the generation of elastic strain. The resulting radius of curvature is denoted by R_1. Then, the sheet can be heated and it will immediately begin to unfold as the transformation strain is recovered. Once the sheet unfolds enough to return to the stowed configuration, it will be in a state similar to the previous scenario when the

transformation strain has just overcome M_0. From this point up until full transformation, both scenarios will behave in the same manner.

For this analysis, M_0 is assumed to be applied fully to the stowed configuration before transformation. This simplifies the analysis by not including a dynamic moment, but is sufficient because this analysis only seeks to determine the steady state radius of curvature after transformation has completed. In order to help describe this situation, Figure 30 presents the different configurations and associated radii of curvature definitions. Note that R_0 will be of the opposite sign because it describes curvature in the opposite direction of the stowed configuration (see Figure 29).

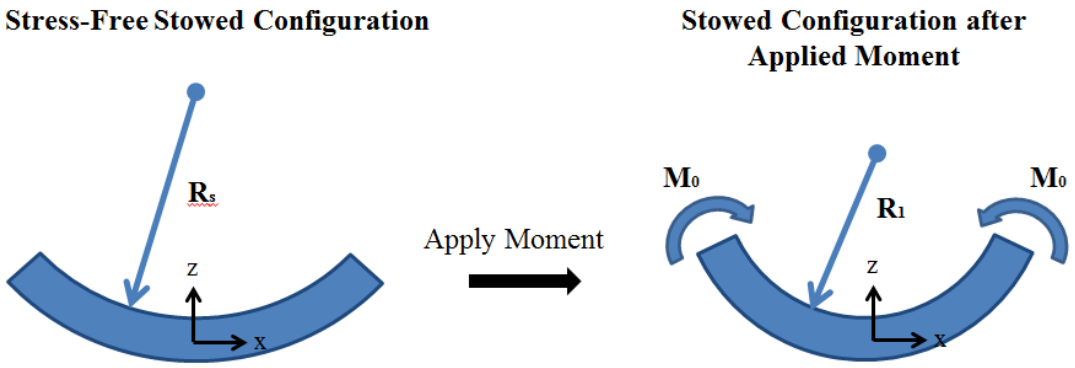

Figure 30. Stowed configuration before and after applied moment.

The stowed configuration is defined fully by the case study parameters through the relation of the radius of curvature to the arc length s and the angle of the fold arc θ:

$$R_s = -\frac{s}{\theta} = -\frac{3W}{\pi}, \qquad (A.17)$$

where the arc length is equivalent to the panel spacing, such that $s = W$, and the angle of the fold arc is $\frac{\pi}{3}$ (this is the angle of the fold arc around a corner of the hexagon at the center of the solar array).

Next, R_1 is needed because it represents the radius of curvature of the sheet just before heating begins. Rearranging Equation 14, R_1 is calculated as follows where the SMA is fully in the martensitic phase ($\xi = 1$):

$$R_1 = \left(\frac{1}{R_s} - \frac{M_0}{EI}\right)^{-1}. \qquad (A.18)$$

The last step required before the final radius of curvature R can be calculated is to determine if plastic strain is generated in folding the SMA sheet to the stowed configuration. This is done by first determining the value of the minimum transformation strain that would be required for no plastic strain to be generated, denoted by H_s^*. By again using Equation 14, with the initial radius of curvature set to R_0 and the final radius of curvature set to R_s, rearranging terms results in the following expression:

$$H_s^* = \left(\frac{1}{R_s} - \frac{1}{R_0}\right)\frac{t_s}{2}. \qquad (A.19)$$

This value determines if plastic strain is generated: if $H_s^* \leq H$, no plastic strain is present in the SMA. Otherwise, the SMA sheet contains plastic strain in the stowed configuration. This plastic strain is only present near the top and bottom faces of the sheet where the pre-strain is larger than H_s. The distance from the centroidal axis at which plastic strain begins is denoted by z^*:

$$z^* = \frac{H_s}{H_s^*}\frac{t_s}{2}. \tag{A.20}$$

For $|z| > |z^*|$, the pre-strain is constant through the thickness, which results in the following updated expression for the pre-strain:

$$\varepsilon^{t0} = \begin{cases} -\dfrac{2zH_s^*}{t_s}, & -z^* \le z \le z^* \\ -\dfrac{2z^*H_s^*}{t_s}, & z > z^* \\ \dfrac{2z^*H_s^*}{t_s}, & z < -z^* \end{cases}. \tag{A.21}$$

By substituting Equation 21 into Equation 6 and solving the integral, M^t is expressed as follows for the SMA in the austenitic state:

$$M^t = LEz^*H^*\left(\frac{z^{*2}}{3t_s} - \frac{t_s}{2}\right). \tag{A.22}$$

Thus, by substituting Equation 22 into Equation 8, setting $M = M_0$, and rearranging terms, the final radius of curvature R is defined as follows:

$$R = \left[\frac{1}{R_1} - \frac{12M_0}{LEt_s^3} - \frac{12z^*H^*}{t_s^3}\left(\frac{z^{*2}}{3t_s} - \frac{t_s}{2}\right)\right]^{-1}, \tag{A.23}$$

where $\frac{d^2w}{dx^2} = \frac{1}{R} - \frac{1}{R_1}$. Note that this equation provides R only if plastic strain is present in the SMA, else the sheet will unfold to the flat configuration ($R = \infty$). Thus, the radius of curvature after actuation is known for any combination of design alternative L_s and

uncertain parameter H_s. The following equation is used to convert the radius of curvature to the fold angle θ:

$$\theta = \pi - \frac{W}{2R}. \qquad (A.24)$$

By assuming that the stresses in the SMA involved in this case study are below the minimum stress to achieve detwinning in the SMA, the sheet will remain at this radius of curvature even after cooling back to martensite (i.e., this is one-way SMA actuation).

9.2 Bending of the SMA Torque Tube Concept

Now consider the SMA torque tube in pure torsion. The analysis is similar to the bending analysis presented previously but with the associated changes to torsion from bending. In this case, it is assumed that the tube undergoes only shear deformation. For a torque tube with a twisted reference configuration, the total strain γ_{xz} is given as follows [72]:

$$\gamma_{xz} = r\frac{d\phi}{dx} = \frac{r(\phi - \phi_0)}{L_t}, \qquad (A.25)$$

where r is the radius of the torque tube, ϕ is the final angle of twist of the torque tube, and ϕ_0 is the initial angle of twist. Although this analysis could be performed without the incorporation of ϕ_0, it is included to mirror the SMA sheet analysis. Assuming that the shear modulus G remains constant throughout SMA transformation and that the only

source of inelastic strain is the transformation strain of the SMA, the shear stress in the tube is given by the following:

$$\tau_{xz} = G(\gamma_{xz} - \gamma^t + \gamma^{t0}), \tag{A.26}$$

where γ^t is the shear transformation strain and γ^{t0} is the initial shear pre-strain. Due to the setup of the two SMA concepts, the applied moment is identical for both the SMA sheet and torque tube. Under only shear stress, the following relation for the moment load is obtained:

$$M = \int_A \tau_{xz} r dA = \frac{GJ(\phi - \phi_0)}{L_t} - T^t, \tag{A.27}$$

$$T^t = \int_A G(\gamma^t - \gamma^{t0}) r dA, \tag{A.28}$$

where J is the polar moment of inertia of the cross-section of the tube. Linear pre-strain through the thickness is assumed:

$$\gamma^{t0} = \frac{H_t r}{r_0}. \tag{A.29}$$

The transformation strain is a function of the martensitic volume fraction ξ as follows:

$$\gamma^t = \xi \gamma^{t0} = \xi \frac{H_t r}{r_0}. \tag{A.30}$$

By incorporating the transformation shear strain fields into T_t in Equation 28, Equation 27 reduces to the following:

$$M = \frac{GJ(\phi - \phi_0)}{L_t} - \frac{GJH_t(\xi - 1)}{r_0}. \quad (A.31)$$

where r_0 is the outer radius of the torque tube. Rearranging this equation yields the angle of twist when the maximum amount of pre-strain is utilized. Similar to the application of the SMA sheet, the analysis will be performed by assuming that M_0 is fully applied before actuation. Further, the same configurations must be defined (from Figure 29 and Figure 30) where each configuration is now defined by an angle of twist instead of a radius (identical subscripts to the SMA sheet are used to define the different configurations). First, the initial configuration is defined by ϕ_0 as follows:

$$\phi_0 = -\frac{M_0 L_t}{GJ}. \quad (A.32)$$

For the SMA torque tube, the stowed configuration is defined by $\phi_s = \frac{\pi}{3}$. Next, ϕ_1 represents the angle of twist of the tube just before heating begins and is calculated as follows where the SMA is fully in the martensitic phase ($\xi = 1$):

$$\phi_1 = \phi_s - \phi_0. \quad (A.33)$$

Again, it must be determined if plastic strain is generated in twisting the SMA torque tube to the stowed configuration. The value of the minimum transformation shear strain that would be required for no plastic shear strain to be generated is denoted by H_t^*.

By rearranging Equation 31 with the initial angle of twist set to ϕ_s and final angle of twist set to ϕ_0, H_t^* is calculated as follows:

$$H_t^* = -\frac{r_0(\phi_0 - \phi_s)}{L_t}. \tag{A.34}$$

Identical to the SMA sheet, if $H_t^* \leq H_t$, no plastic strain is present in the SMA tube. Otherwise, the tube contains plastic strain in the stowed configuration. In this case, plastic strain is only present near the outer edge of the torque tube where the pre-strain is larger than H_t, but not near the inner edge. This is because the maximum strain occurs at the outer edge of the tube. The radius at which plastic strain begins is denoted by r^*:

$$r^* = \frac{H_t r_0}{H_t^*}. \tag{A.35}$$

For $r > r^*$, the transformation strain is constant, resulting in the following expression:

$$\gamma^{t0} = \begin{cases} \dfrac{H_t^* r}{r_0}, & r \leq r^* \\ \dfrac{H_t^* r^*}{r_0}, & r > r^* \end{cases}. \tag{A.36}$$

By substituting Equation 36 into Equation 28 and solving the integral, T^t is calculated as follows for the SMA in the austenitic state:

$$T^t = -\frac{G H_t^* \pi}{r_0}\left(\frac{r_0^4 - r_i^4}{2} + \frac{2r^*(r_0^3 - r_i^3)}{3}\right). \tag{A.37}$$

Thus, by substituting Equation 37 into Equation 27, setting $T = M_0$, and rearranging terms, the final angle of twist ϕ is defined as follows:

$$\phi = \phi_1 + \frac{2L_t}{G\pi(r_0^4 - r_i^4)}\left[M_0 + \frac{GH_t^*\pi}{r_0}\left(\frac{r_0^4 - r_i^4}{2} + \frac{2r^*(r_0^3 - r_i^3)}{3}\right)\right]. \qquad (A.38)$$

Again, note that this equation provides ϕ only if plastic strain is present in the SMA, else the sheet will unfold to the flat configuration ($\phi = 0$). Thus, the fold angle after actuation is known for any combination of design alternative L_t and uncertain parameter H_t. Again, this is one-way actuation such that the array remains at this angle after cooling to martensite.

9.3 Applied Power for the SMA Concepts

In order to determine the applied power required to heat the SMA concepts, the latent heat of the SMA must be incorporated. To do so, the local form of the heat equation for SMA's must be considered under reverse transformation [73]. Assuming thermal expansion is negligible, adiabatic natural boundary conditions, and a uniform temperature, the heat equation reduces to the following:

$$\rho_{SMA} c_p \dot{T} + (\rho_{SMA}\Delta s_0 T - \pi_{rev}^t)\dot{\xi} = \rho_{SMA} r_V, \qquad (A.39)$$

where ρ_{SMA} is the density of the SMA, T is the temperature, s_0 is the specific entropy, π_{rev}^t is the thermodynamic driving force for transformation, and r_V is the heat generation per unit mass. The transformation surface Φ_{rev}^t is zero during reverse transformation and is given by the following [70]:

$$\Phi_{rev}^{t} = -\pi_{rev}^{t} - Y_{rev}^{t}, \tag{A.40}$$

where Y_{rev}^{t} is the critical thermodynamic driving force. By incorporating Equation 40 during reverse transformation and integrating Equation 39 over time where the SMA is already in the fully austenitic phase, the total applied power required to heat the SMA can be obtained:

$$\int_0^{t_0} \rho_{SMA} r_V dt = \rho_{SMA} r_V t_0 = \int_0^{t_0} \left[\rho_{SMA} c_p \dot{T} + (\rho_{SMA} \Delta s_0 T - \pi_{rev}^{t})\dot{\xi}\right] dt, \tag{A.41}$$

$$\rho_{SMA} r_V t_0 = \int_0^{t_0} \rho_{SMA} c_p \dot{T} dt + \int_{t_1}^{t_2} (\rho_{SMA} \Delta s_0 T + Y_{rev}^{t})\dot{\xi} dt, \tag{A.42}$$

where t_0 is the total time for which the SMA has power applied, t_1 and t_2 are the times between 0 and t_0 when the SMA begins and ends transformation. Using Lagrange polynomials, T and ξ are interpolated in time as follows:

$$T = \frac{t_0 - t}{t_0} T_0 + \frac{t}{t_0} T_f, \tag{A.43}$$

$$T = \frac{t_2 - t}{t_2 - t_1}\left(A_s + \frac{\sigma_1}{C^A}\right) + \frac{t - t_1}{t_2 - t_1}\left(A_f + \frac{\sigma_2}{C^A}\right), \tag{A.44}$$

$$\xi = \frac{t_2 - t}{t_2 - t_1}, \tag{A.45}$$

where T_0 is the initial temperature of the SMA, T_f is the final temperature of the SMA, A_s is the austenitic start temperature, A_f is the austenitic finish temperature, C^A is the

stress influence coefficient of austenite, and σ_1 and σ_2 are the von Misses stresses at $t = t_1$ and $t = t_2$, respectively. Equation 43 provides the temperature over the time intervals $0 \leq t \leq t_1$ and $t_2 \leq t \leq t_0$, and Equations 44 and 45 provide the temperature and martensitic volume fraction over the time interval $t_1 \leq t \leq t_2$.

From Equations 43 and 44, the temperature and martensitic volume fraction time rates are expressed as follows:

$$\dot{T} = -\frac{1}{t_0}T_0 + \frac{1}{t_0}T_f = \frac{T_f - T_0}{t_0}, \quad (A.46)$$

$$\dot{\xi} = -\frac{1}{t_2 - t_1}. \quad (A.47)$$

By substituting Equations 43-47 into Equation 41, symbolic integration generates the following expression:

$$\rho_{SMA} r_V t_0 = \rho_{SMA} c_p (T_f - T_0) - \frac{1}{2}\rho_{SMA}\Delta s_0 \left(\frac{(\sigma_1 + \sigma_2)}{C^A} + A_s + A_f\right) - Y_{rev}^t, \quad (A.48)$$

where Y_{rev}^t is assumed to be independent of σ (i.e., $C^A \approx C^M$) and the von Misses stress is assumed to evolve linearly from σ_1 to σ_2 during transformation. For the sheet and torque tube, the von Misses stresses are purely an elastic response to the applied moment. Combining Equations 4 and 25 with Equations 16 and 32, respectively, the von Misses stress for the sheet and torque tube are expressed as follows:

$$\sigma_{1,s} = \sigma_{2,s} = |\sigma_{xx}| = \left|\frac{12 M_0 z}{L_s t_s^3}\right|, \quad (A.49)$$

$$\sigma_{1,t} = \sigma_{2,t} = |\tau_{xz}| = \left|\frac{2M_0 r}{\pi(r_0^4 - r_i^4)}\right|. \quad (A.50)$$

Thus, the total energy required can be obtained through the volume integration of Equation 48. Assuming a uniform temperature field, the energy required for the SMA sheet E_s and SMA torque tube E_t are calculated as follows:

$$E_s = V_s \rho_{SMA} r_V t_0$$
$$= V_s \left[\rho_{SMA} c_p (T_f - T_0) - \frac{1}{2}\rho_{SMA}\Delta s_0 \left(\frac{6M_0}{L_s t_s^2 C_A} + A_s + A_f\right) - Y_{rev}^t\right], \quad (A.51)$$

$$E_t = V_t \rho_{SMA} r_V t_0$$
$$= V_t \left[\rho_{SMA} c_p (T_f - T_0) - \frac{1}{2}\rho_{SMA}\Delta s_0 \left(\frac{8M_0 L_t (r_o^3 - r_i^3)}{3C^A V_t (r_o^4 - r_i^4)} + A_s + A_f\right) - Y_{rev}^t\right], \quad (A.52)$$

where V_s and V_t are the volumes of the SMA sheet and SMA torque tube, respectively. By assuming a constant maximum transformation strain, no stress dependence on the critical thermodynamic driving force, and all hardening function exponents equal to 1, $\rho_{SMA}\Delta s_0$ and Y_{rev}^t reduce to the following [3]:

$$\rho_{SMA}\Delta s_0 = -C^A H, \quad (A.53)$$

$$Y_{rev}^t = \frac{\rho_{SMA}\Delta s_0 (M_s + M_f - A_s - A_f)}{4} = -\frac{C^A H (M_s + M_f - A_s - A_f)}{4}, \quad (A.54)$$

where M_s is the martensitic start temperature and M_f is the martensitic finish temperature. By substituting Equations 53 and 54 into Equations 51 and 52, the required energies are expressed as follows:

$$E_s = V_s\left[\rho_{SMA}c_p\left(T_f - T_0\right) + \frac{1}{2}C^A H_s\left(\frac{6M_0}{L_s t_s^2 C_A} + A_s + A_f\right) + \frac{C^A H_t\left(M_s + M_f - A_s - A_f\right)}{4}\right], \quad (A.55)$$

$$E_t =$$

$$V_t\left[\rho_{SMA}c_p\left(T_f - T_0\right) + \frac{1}{2}C^A H_t\left(\frac{8M_0 L_t(r_o^3 - r_i^3)}{3C^A V_t(r_o^4 - r_i^4)} + A_s + A_f\right) + \frac{C^A H_t\left(M_s + M_f - A_s - A_f\right)}{4}\right]. (A.56)$$